The Dissertation Committee for Guohua Ma
Certifies that this is the approved version of the following dissertation:

Topological Consistency in Skeletal Modeling with Convolution Surfaces for Conceptual Design

Committee:

Richard H. Crawford, Supervisor

Matthew Campbell

Billy V. Koen

Kristin L. Wood

Warren N. Waggenspack, Jr.

Topological Consistency in Skeletal Modeling with Convolution Surfaces for Conceptual Design

by

Guohua Ma, B.S.; M.S.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

December, 2007

Dedicated to my family

Acknowledgements

This dissertation could not be finished without the help and support of many people who are gratefully acknowledged here.

At the very first, I would like to gratefully and sincerely thank my advisor, Dr. Richard Crawford for his guidance, understanding, and patience during my graduate study. He guided me into the topology field and provided enough flexibility for me to maintain my commitments to my family. This kindness and the opportunities that Dr. Richard Crawford continually provided to me are a gift I truly treasure.

I would like to express my thanks for all other members of my doctoral committee for their invaluable comments and suggestions. Thanks to Dr. Matthew Campbell for his comments about the computing time discussed in this dissertation. I appreciate Dr. Billy V. Koen for his critical comments on topology. I especially thank him for supporting me as his teaching assistant for many years. I appreciate Dr. Kristin Wood for his comments; I especially thank him for supporting me in the second year of my graduate study. Thanks to Dr. Waggenspack for his suggestions on convolution surfaces.

I would like to thank my co-TAs Dr. Zhaohong Wu and Aravindh Johendran for their support and wonderful teamwork regarding the ME205 class for many years. Thanks are also given to MELRC staff Peggy Berry, Susan Ponder, and Cengiz Vural for their kindly and continuous support for my work as a TA.

I would like to thank many friends in the MADLAB, including Dr. Matthew Green, Amelia Palmer Hansen, Edward Huang, and Aparajit Pratap. It won't be as much fun without them. I especially thank Dr. Monty Greer and Dr. Mike Van Wie, who helped me prepare for the oral qualifying exam.

Ms. Cindy Raman, Ms. Ruth Schwab, Ms. Jessica Miller, Ms. Diane Selken, and Ms. Becky Harrison-Ray, and Mr. Erik Felthauser from the Department of Mechanical Engineering are especially thanked for their care and help.

Friends outside of school have been an essential stabilizing force and made my life in Austin much easier. Many thanks are given to Yongtong Lu, Dr. Ning Liu, Tielin Cui, Haiqing Zhang, Dr. Xiuli Gai, and Yuping Liang for their generous help with my daughter these years.

Finally, my family is the root that feeds my growth – all of them believed in me and cheered me up through many hard years. I cannot thank them all enough for their love, care, and support. Many thanks are given to my parents for their understanding, and love. Thanks to my lovely daughter Chelsea, who brought endless happiness to me these years. Finally, thank you to my husband Fu Zhang, whose love has encouraged me through many hard times.

Topological Consistency in Skeletal Modeling with Convolution Surfaces for Conceptual Design

Publication No._____

Guohua Ma, Ph.D.

The University of Texas at Austin, 2007

Supervisor: Richard H. Crawford

This dissertation describes a new topology analysis tool for a skeletal based geometric modeling system for conceptual design. Skeletal modeling is an approach to creating solid models in which the engineer designs with lower dimensional primitives such as points, lines, and triangles. The skeleton is then "skinned over" to create the surfaces of the three-dimensional object. In this research, convolution surfaces are used to provide the flesh to the skeleton. Convolution surfaces are generated by convolving a kernel function with a geometric field function to create an implicit surface. Certain properties of convolution surfaces make them attractive for skeletal modeling, including: (1) providing analytic solutions for various geometry primitives (including points, line segments, and triangles); (2) generating smooth surfaces; and (3) providing well-behaved blending. We assume that engineering designers expect the topology of a skeletal model to be identical to that of the underlying skeleton. However, the topology of convolution surfaces can change arbitrarily, making it difficult to predict the topology of the generated surface from knowledge of the topology of the skeleton.

To address this issue, we apply Morse theory to analyze the topology of convolution surfaces by detecting the critical points of the surfaces. We developed an efficient and intelligent algorithm to find the critical points (CPs) by analyzing the skeleton.

The critical points provide valuable information about the topology of the convolution surfaces. By tracking the CPs, we know where and what kind of topology changes happen when the threshold value reaches the critical value at the CP. Topology matching is done in two steps: (1) global topology is tested by comparing the Betti numbers (number of component, loops, and voids) of the skeleton and the generated convolution surfaces; (2) with matched Betti numbers, local topology is tested by comparing the location of each loop and void area between the skeleton and surfaces. If the topology does not match, appropriate heuristics for determining parameter values of the convolution surfaces are applied to force the surface topology to match that of the skeleton. A recommend threshold value is then provided to generate the topology matched convolution surfaces.

Table of Contents

List of Tables xiii

List of Figures xiv

Chapter 1 Introduction 1

1.1 Conceptual Mechanical Design 1

1.2 Skeleton Based Shape Modeling 4

1.3 Problem Statement 5

1.3.1 Skeletal Shape Modeling 5

1.3.2 Topology Problems 9

1.4 Contributions 11

1.5 Outline of the Remaining Chapters 12

Chapter 2 Background and Previous Work on Surface Reconstruction 14

2.1 Shape Modeling from Skeletons 14

2.1.1 Parametric Representation 14

2.1.2 Simplicial Representations 16

2.1.3 Implicit Representations 19

2.2 Convolution Surfaces 23

2.2.1 Background 23

2.2.1.1 Kernel Function 24

2.2.1.2 Field Function 25

2.2.1.3 Properties of Convolution Surfaces 26

2.2.2 Previous Work on Convolution Surfaces 28

2.2.3 Unwanted Blending 30

2.3 Summary 31

Chapter 3 Background and Literature on Surface Topology 33

3.1 Topology 33

3.1.1 Topology Invariants 34

3.1.1.1 Euler Characteristic 34

3.1.1.2 Betti Numbers 34
3.1.2 Characterizing Topology 35
3.2 Morse Theory 37
3.2.1 Previous Work on Morse Theory 39
3.2.2 Critical Points 40
3.2.3 Detecting Critical Points 41
3.3 Matching Topology 42
3.4 Summary 43
Chapter 4 Searching for Critical Points of the Convolution Surfaces 45
4.1 Searching Space 45
4.1.1 Interpreting the Designer's Intent 46
4.1.2 Heuristics for Locating Critical Points of Convolution Surfaces. 47
4.2 Algorithm 51
4.2.1 Partitioning the Space 51
4.2.1.1 Convex Hull 52
4.2.1.2 Identify Polygons 53
4.2.1.3. Polygon Partition 55
4.2.2 Detailed Algorithm 57
4.2.2.1 Searching for Index-2 CPs 57
4.2.2.2 Searching for Index-1 CPs 59
4.3 Searching Algorithm 60
4.3.1 Problem Definition 60
4.3.2 Sequential Quadratic Programming (SQP) 61
4.3.3 Computing Time 63
4.4 Case Studies 63
4.4.1 Example One 63
4.4.2 Example Two 67
4.5 Summary 70

Chapter 5 Topology Analysis of Convolution Surfaces 71
5.1 Topological Invariants 71
5.2 Topology of the Skeleton 72
5.3 Topology Matching 73
5.4 Matching the Topology of Convolution Surfaces and Skeleton 80
5.4.1 Distinguishing Different Topologies 81
5.4.2 Modifying the Skeleton and its Properties 88
5.4.3 Parameter *s* – Width of the Cauchy Kernel Function 89
5.4.4 Adjusting the Value of *s* 91
5.5 Summary 93

Chapter 6 Case Studies 95
6.1 Bottle Opener 95
6.1.1 Case 1 96
6.1.2 Case 2 102
6.2 Tetrahedral Elements 104
6.3 Design of a Steering Wheel 109
6.4 Computing Time 115
6.5 Summary 116

Chapter 7 Conclusion and Future Work 117
7.1 Contributions 118
7.2 Future Work 119
7.3 Applications 123

Appendix A Convolution Surface Field Function [Shre98] 128
A.1 Point sources 128
A.2 Line segments 129
A.3 Arcs 130
A.4 Triangles 131
A.5 Plane 133

Bibliography 134

Vita 143

List of Tables

Table 3–1. Relationship between CP type and topology changes. 39

Table 4–1. Geometry of the skeleton for example one. 64

Table 4–2. CPs of the convolution surfaces from example one. 66

Table 4–3. Geometric information for the skeleton of example two. 67

Table 4–4. CPs of the convolution surfaces for the ring.....68

Table 5–1. CPs of the convolution surfaces.....76

Table 5–2. Topology changes of convolution surfaces. 80

Table 5–3. The geometry information and the value of *s*. 87

Table 6–1. Geometry information for the bottle opener. 96

Table 6–2. Critical values of the bottle opener. 98

Table 6–3. Time for computing CPs for different cases. 115

List of Figures

Figure 1–1. Block diagram of design process [Fren99].....3

Figure 1–2. Skeleton and convolution surfaces. 7

Figure 1–3. Longhorn logo and its skeleton. 8

Figure 1–4. Longhorn generated with convolution surfaces. 8

Figure 1–5. Exhaust manifold generated from skeletal model. 9

Figure 2–1. Convolution surface and its components [Sher99].....24

Figure 2–2. Cauchy kernel.....25

Figure 2–3. Example convolution surfaces of a line segment [Sher99]. 25

Figure 2–4. Superposition property of convolution surface. 27

Figure 2–5. Region of influence. 28

Figure 2–6. Unwanted blending [AJC02].....31

Figure 4–1. Two different dumbbell skeletons and the resulting convolution surface. 47

Figure 4–2. Three point configurations with Voronoi diagrams and Delaunay triangulations [Sier99]. 49

Figure 4–3. Corresponding convolution surfaces of the skeletons shown in Figure 4–2.....49

Figure 4–4. An example of the wrapping algorithm.....52

Figure 4–5. Halfedge data representation. 53

Figure 4–6. Halfedge data structure implementation.....54

Figure 4–7. Partition of a polygon 56

Figure 4–8. Searching for index–2 CPs. 58

Figure 4–9. Searching for index–1 CPs. 59

Figure 4–10. An example skeleton and its resulting partition. 64
Figure 4–11. CPs of the convolution surfaces from example one. 66
Figure 4–12. Skeleton and CPs of example two. 68
Figure 4–13. Generated convolution surfaces with different threshold values *T*. .69
Figure 5–1. Algorithm to calculate the topology of composite convolution surfaces. 74
Figure 5–2. An example of topology matching. 75
Figure 5–3. Topology analysis of example in Figure 5–2. 77
Figure 5–3 (continued). Topology analysis of example in Figure 5–2. 78
Figure 5–4. Relationship between topology changes and threshold value *T*. 79
Figure 5–5. Case 1: The skeleton and its CP locations. 82
Figure 5–6. Results from analyzing case 1. 83
Figure 5–7. Results from analyzing case 2. 84
Figure 5–8. Topology match analysis for case 2. 86
Figure 5–9. Skeleton and CPs for case 3. 87
Figure 5–10. Generated convolution surfaces. 87
Figure 5–11. Relationship between field function and width *s*. 90
Figure 5–12. Skeleton of two close line segments and their widths. 91
Figure 5–13. Field function for two line segments close to each other. 92
Figure 5–14. Modified convolution surfaces for skeleton in Figure 5–8. 93
Figure 6–1. Skeleton of the bottle opener. 96
Figure 6–2. Partitioned bottle opener skeleton and CPs of generated convolution surfaces. 97
Figure 6–3. Field functions for line segment skeletons with different configurations at *z*=0 plane. 99

Figure 6–4. Loop in the bottle opener skeleton. 100
Figure 6–5. Topology changes for the bottle opener for different threshold values.. 101
Figure 6–6. The relationship between *T* and *s*. 103
Figure 6–7. Two different configurations of the bottle opener. 104
Figure 6–8. The convolution surfaces generated based on four triangles formed into a tetrahedral element. 105
Figure 6–9. Geometry of the tetrahedron element. 106
Figure 6–10. Convolution surfaces of simulated tetrahedron with different values of *T*. 107
Figure 6–11. Index-2 CPs for example tetrahedron. 108
Figure 6–12. Convolution surfaces of simulated tetrahedron after adding one sphere. 109
Figure 6–13. Skeletons of two steering wheel designs. 110
Figure 6–14. Steering wheel shapes that do not match the skeleton topology. ...112
Figure 6–15. Final shapes of the two steering wheel designs. 112
Figure 6–16. Different shapes for different values of *T* for design 2. 113
Figure 6–17. Images of two steering wheel designs. 114
Figure 7–1. Non-circular cross section shape. 121
Figure 7–2. Detail of cross section of shape in Figure 7–1. 121
Figure 7–3. Stress analysis of a conceptual clamp design. 125
Figure 7–4. Use of convolution surfaces to model blood cells. 127
Figure A–1. Line primitive [Sher98]. 129
Figure A–2. Arc primitive [Sher98]. 130
Figure A–3. Triangle primitive [Sher98]. 131

Chapter 1 Introduction

This dissertation describes a new topology analysis tool for a skeletal based conceptual design system. Skeletal modeling is an approach to creating solid models in which the engineer designs with lower dimensional primitives such as points, lines, and triangles. The skeleton is then "skinned over" to create the surfaces of the three dimensional object. In this research, we use convolution surfaces to flesh out the skeleton. Convolution surfaces are generated by convolving a kernel function with a geometric field function to create an implicit surface. Certain properties of convolution surfaces make them attractive for skeletal modeling, including: (1) providing analytic solutions for various geometry primitives (including points, line segments, and triangles); (2) generating smooth surfaces; (3) and providing well-behaved blending. We assume that engineering designers expect the topology of a skeletal model to be identical to that of the underlying skeleton. When we talk about topology, usually we mean how the space is connected. We propose that the topology of a design can be represented by the skeleton. But the final design will be a more complicated smooth surface. Proper parameter values must be carefully selected to ensure the topology of the generated convolution surfaces matches that of the root skeleton.

1.1 CONCEPTUAL MECHANICAL DESIGN

At its most abstract level, the product development process can be characterized by three phases: understand the opportunity, *develop a concept*, and implement a concept. The concept development phase encompasses all activities leading to decisions about the functionality and general form of the product [OW00].

A model of the design process is shown in Figure 1–1. Conceptual design is an early design stage in which various alternative design concepts are formulated, and one (or a few) is chosen according to product requirements for embodiment. A concept variant (an alternative product concept) includes functions, possible behaviors, form/structure (layout), and associated properties. Properties include materials, assembly-level tolerances, critical surface roughness and hardness parameters, and critical dimensions. Detailed product information is derived from the conceptual design. It is generally agreed that the most critical stage of the design process is conceptual design. At this stage, many of the required resources are allocated. A commitment to a particular product concept is made, and it is often difficult (expensive) to reverse this commitment.

Geometric design is one of the most important aspects of conceptual design. It focuses on specifying the basic geometric structure and functionality of the product. Less attention is paid to details. Geometric design also determines the overall appearance of the product. Appearance can have a great influence on buying decisions of customers, who are significantly guided by their perception of shape. Thus, the shape of a product such as a car or a household appliance plays an important role in the success or failure of the product [WS01]. Generation of the initial appearance of the product at the early conceptual design stage can provide guidance for understanding the design and for design improvements; and can greatly reduce the development time and cost.

In this research, we focus on geometric design in the conceptual design stage. Currently computer-aided design (CAD) systems are widely used to shorten the design to manufacturing cycle time. However, most CAD systems are detail design oriented. These CAD systems use modeling methods that represent the surfaces of an object explicitly as sets of polygons or parametric patches and are difficult for use in conceptual design, as they require a complete geometric specification. At the conceptual design stage, the

geometry information is often incomplete and thus it is hard to use commercial CAD systems for conceptual modeling.

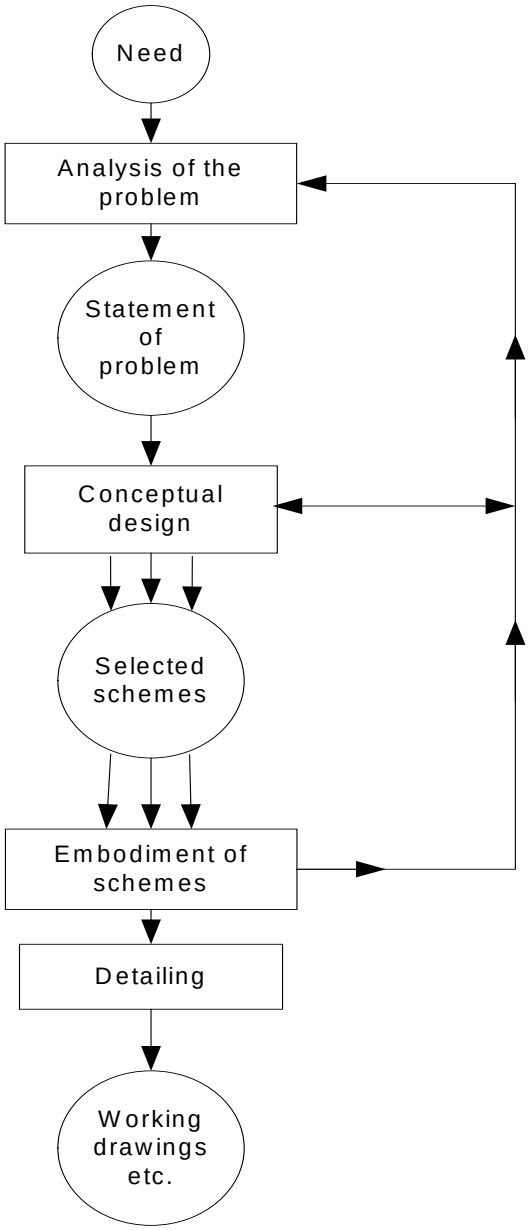


Figure 1–1. Block diagram of design process [Fren99].

1.2 SKELETON BASED SHAPE MODELING

A *skeleton* is a structured set of geometric primitives (usually points, segments or polygons) that suggests the shape of an object and from which smooth surfaces can be generated. We propose that the topology of a design can be represented by the skeleton. But the final design will be a more complicated smooth surface. Thus, the problem of generating a smooth surface model from the skeleton arises.

Most existing skeletal modeling tools generate the shape based on a skeleton formed by simple primitives such as points. Numerous reconstruction methods have been developed over the past two decades to visualize 3D surfaces defined by a set of points in space, including those described in [HDD92, BBX97, BBCS99, SCDL00, EM94, Edel92]. Depending on the form of the surface, the shapes generated can be discontinuous when relatively few skeletal elements are used to define the shape. To allow richer geometric/topological coverage, we claim more complicated geometric primitives such as lines and triangles should be used to construct the skeleton.

Skeletal models provide a convenient method for defining implicit surfaces. A number of different implicit surface forms have been used to generate shape from skeletons, including [BS91, SM98, Sher99, JTFP01, MJFP01, OP04]. The convolution surface is a type of implicit surface obtained by convolving a kernel along a skeleton. Convolution surfaces incorporate smooth blending and easy manipulability of potential surfaces while expanding the available skeletal elements from points to lines, polygons, planar curves and regions, and in principle, any geometric primitive [Bloo97]. Generally, convolution surface methods produce circular cross-sectioned shapes due to the nature of the kernel function. Convolutions surfaces are the focus of this research.

1.3 PROBLEM STATEMENT

Develop an algorithm to ensure that the topology of a skeleton based convolution surface is consistent with the topology of its skeleton to support conceptual design. The problem statement is twofold: i) a modeling tool is needed to generate smooth surfaces based on the skeleton, and ii) the topology of the generated surfaces must agree with that of the skeleton to maintain those aspects of the designer's intent that are related to topology, including aspects of product functionality and appearance.

1.3.1 Skeletal Shape Modeling

Thompson [Thom00] developed a skeletal modeler that generates surfaces based on a union of balls. The current research is an extension of his work. Surfaces generated by a union of balls are not necessarily pleasingly smooth, as they tend to look like spheres that have been set-unioned, as the name implies. We propose to improve the smoothness of the surfaces generated from the skeleton. Several questions must be addressed to improve the quality of the surfaces:

(1) What other methods can be used to generate the smooth surfaces?

(2) How can smooth surfaces be constructed for a variety skeletal elements?

(3) How can we measure whether the generated surfaces have the same topology as the skeleton?

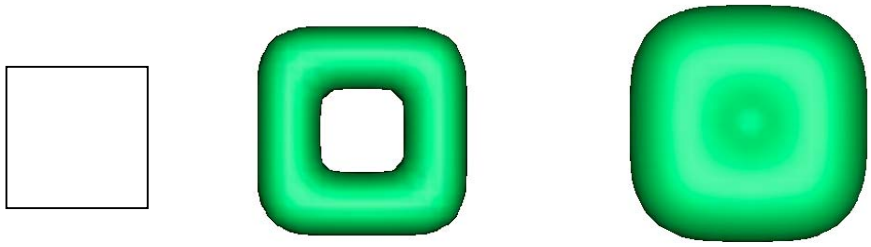
Based on reviewing different methods to generate the surfaces that are documented in the literature[1], convolution surfaces stand out as particularly attractive for this application. Again, the appealing properties of convolution surfaces include generating smooth surfaces, automatic blending, analytic solutions to many skeletal primitives, and available visualization algorithms.

[1] A complete literature review is contained in Chapter 2.

The second question concerns surface generation for various skeletal elements. The method used in this dissertation is introduced by Bloomenthal [BS91]. The kernel function used is based on Shersyuk's Cauchy function[2] [Sher98, Sher99]. This work provides analytic solutions to various skeletal elements with the Cauchy kernel. The research reported in this dissertation is focused on how to control the shape based on the skeleton and the parameters of the kernel function. Careful control of parameter selection is required to provide surfaces that match the designer's intent with respect to topology.

Modeling with convolution surfaces guarantees a continuous and smoothly blended surface, as the shape corresponds to a coating of the skeleton. This is an advantage for conceptual design. However, this property also turns out to be a problem. When two skeleton elements are placed too close to each other and they are blended, there is little control over the topology of the final shape. By topology, we mean how the space is connected. A simple example is shown below in Figure 1–2. Figure 1–2(a) shows the skeleton, which is a loop formed by four line segments. Figures 1–2(b) and (c) are convolution surfaces generated based on the skeleton using different values of certain parameters. Figure 1–2(b) has a loop and thus has the same topology as the skeleton. In Figure 1–2(c), the loop is filled and the topology of the surface is different from that of the skeleton.

[2] Please refer to Chapter 2 for details.



(a) skeleton (b) surface (same topology) (c) surface (different topology)

Figure 1–2. Skeleton and convolution surfaces[3].

Each feature of an object has its special requirement and/or its own functionality. The skeleton should reflect these requirements. We propose that the skeleton represents not only the topology of the convolution surfaces, but aspects of the shape and functionality of the final design. For example, the handle of a mug is used to grip the mug. A skeleton of the mug should clearly show that intent, and the generated convolution surfaces should represent the handle as well. If the generated surfaces fill the loop representing the handle, the functionality related to holding disappears. Hence topology is a key issue in generating the final set of surfaces.

Two examples using convolution surfaces are shown below.

- UT Longhorn

The longhorn is the logo of The University of Texas at Austin. The skeleon in Figure 1–3 is drawn based on the longhorn logo. The skeleton is formed by a collection of line segments and triangle elements. Based on the skeleon, the convolution surfaces, shown in Figure 1–4, were generated with the Cauchy kernel. The surfaces are smooth and well blended.

[3] The images of convolution surfaces in this dissertation were rendered with Hyperfun [webHyer07].

Figure 1–3. Longhorn logo and its skeleton.

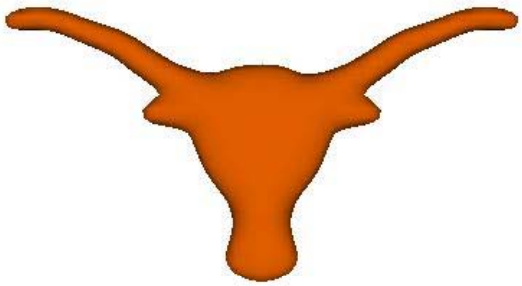


Figure 1–4. Longhorn generated with convolution surfaces.

- Exhaust Manifold

Most engineering designs involve a multitude of considerations, including strength, deflection, weight, size and shape, wear, lubrication, corrosion, frictional forces, safety, ecology, reliability, etc [Mars87]. Constraints based on these engineering issues influence the shape of a skeletal representation of a design. Geometric constraints are an important class of constraints. When designing a system, space is limited for each part. For example, consider an exhaust manifold used in an automobile (Figure 1–5). The port locations are fixed. The exhaust manifold must fit within a limited spacial envelope in the automobile. We propose that the designer can construct the skeleton of the exhaust

manifold based on such constraints. Then a surface model of the exhaust manifold can be generated by convolution surfaces. In the example showed in Figure 1–5, the topology is a very important issue. The manifold is designed to connect ports 1 and 4 into one duct, and ports 2 and 3 into another duct. Finally the manifold combines these two ducts into a single output duct. The topology of the exhaust manifold must agree with the topology of the skeleton. Otherwise the exhaust gases will not flow properly.

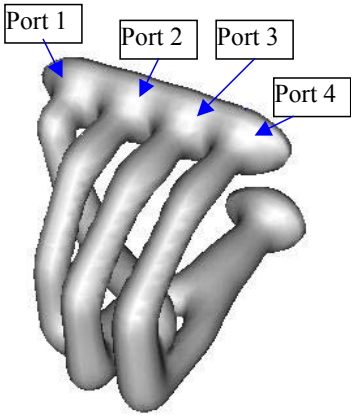


Figure 1–5. Exhaust manifold generated from skeletal model.

1.3.2 Topology Problems

Topology is an abstraction of certain geometrical ideas such as “continuity” and “closeness” [FG01]. Topology is the mathematical study of properties of surfaces that are preserved through deformations, other than tearing [Weis07]. For example, a sphere is topologically equivalent to an ellipsoid (since a sphere can be changed into an ellipsoid by stretching).

A formal definition of topology in terms of set operations is given by [GG99]. Let X be a set. A family T of subsets of X is a topology for X if T has the following properties:

(i) Both X and the empty set belong to T.

(ii) Any union of sets in T belongs to T.

(iii) Any finite intersection of sets in T belongs to T.

A *topology space* is a pair *(X, T),* where X is a set and T is a *topology* for X. The subject of topology consists of several different branches, such as point set topology, algebraic topology, and differential topology. Point-set topology, also called general topology, is the study of the general abstract nature of continuity or "closeness" on spaces. Algebraic topology is the study of intrinsic qualitative aspects of spatial objects (e.g., surfaces, spheres, etc.) that remain invariant under bi-directional continuous one-to-one (homeomorphic[4]) transformations. Differential topology is the field dealing with differentiable functions on differentiable manifolds[5]. It arises naturally from the study of the theory of differential equations. In general, the topology of a surface refers to the connectedness of the shape, which includes the number of disconnected components, the number of tunnels and the number of voids in each component.

In this dissertation, we are interested in the topology of the skeleton and the generated surfaces. The generated surfaces are compact, orientable surfaces. The topology of both the skeleton and the composite convolution surfaces is defined as the number of components, the number of loops or tunnels for the object, and the number of voids inside the object. Thus the composite convolution surfaces should have the same number of components, loops, and voids as that of the skeleton. Certain questions related to the topology arise:

---

[4] Two objects are homeomorphic if they can be deformed into each other by a continuous, invertible mapping. Homeomorphism is equivalent to topological isomorphism.

[5] A manifold is an abstract mathematical space in which every point has a neighborhood that resembles Euclidean space.

(1) How do we calculate the topology of the convolution surfaces?

(2) How can we compare the topology of the surfaces to the underlying skeleton?

(3) How can we modify the convolution surfaces to ensure a topological match?

To answer the first question, we employ Morse Theory, a key tool for studying the topology of spaces. The theory was first introduced by Milnor [Mil63] as a tool to analyze the topology of a continuous function, and was extended by Hart [Hart98] to analyze the topology of implicit surfaces. Morse theory relies on identifying the type of each non-degenerate critical point (CP) of a surface, and analyzing the function value at each CP. Stander and Hart [SH98] used interval analysis and Newton's method to search for the critical points. Their skeleton is formed by points. In our system, the skeleton is formed by higher dimensional skeletal elements (lines, triangles, etc.), which represent the topology of the convolution surfaces. Thompson [Thom00] maintained the topology of the solid model by allowing limited blending of neighboring skeletal segments that does not change the model topology. However, this method cannot be used for convolution surfaces. We use the skeleton's topology information to analyze the offset surfaces.

The second question deals with determining whether the topology of the convolution surfaces matches the root skeleton. Knowing the topology of the convolution surfaces and that of the skeleton, the topology properties can be compared to determine whether the surfaces must be modified to match the topology of the skeleton.

The last question addresses modification of the convolution surfaces to ensure a topological match. Certain rules are applied here and critical parameters of the convolution surfaces are adjusted to create the topological match.

1.4 CONTRIBUTIONS

The contributions of this research include:

- Use of convolution surfaces as a modeling tool to generate smooth surfaces for a skeletal modeling system for conceptual design. Currently analytic solutions for points, line segments, curves, arcs, triangles and planes are known. An approach to simulating tetrahedral elements is also discussed.
- A method for analyzing the topology of convolution surfaces with geometric primitives other than points. Previous research reports only topology analysis for point based implicit surfaces. Morse theory is applied to analyze the topology of convolution surfaces.
- An effective algorithm for identifying the critical points of convolution surfaces by analyzing the geometry of the underlying skeleton. We propose that the skeleton represents not only the final shape of the part, but also the desired topology of the part. By partitioning the skeleton into convex subregions, the CPs can be identified.
- A strategy to match the topology of convolution surfaces with the root skeleton. The shape generated by the convolution surfaces is controlled by adjusting appropriate parameters to maintain topological agreement during manipulation of the skeleton.

1.5 OUTLINE OF THE REMAINING CHAPTERS

The remainder of this dissertation is organized as follows:

- Chapter 2 reviews surface generation methods for skeleton based modeling, including parametric, simplicial, and implicit representations. Constructing convolution surfaces from various skeletal elements is also discussed.
- Chapter 3 discusses the fundamentals of surface topology, and Morse theory. Work related to identifying surface topology and applying Morse theory to analyze surface topology is reviewed.

- Chapter 4 describes the algorithm for locating the critical points of convolution surfaces. The geometry of the skeleton is used to guide the search by partitioning the space into convex subregions, and the sequential quadratic programming (SQP) algorithm is applied to locate the CPs.
- Chapter 5 describes the algorithm for identifying topology changes of the convolution surfaces, as well as the heuristics for matching the topology of the convolution surfaces with that of the skeleton. Topological consistency of the generated surfaces and the root skeleton does not necessarily mean that the generated surfaces reflect the designer's intent. Local geometry should be considered as well. A strategy for modifying the convolution surfaces is also discussed.
- Chapter 6 presents several case studies to illustrate the use of the algorithm presented in this dissertation. A realistic conceptual design example is discussed to show how the proposed method can help a designer early in the design process.
- Finally, Chapter 7 draws conclusions from this research. Limitations of the work are also discussed, as are ideas for extensions and future work.

Chapter 2 Background and Previous Work on Surface Reconstruction

The subject of surface reconstruction has attracted many researchers in the past three decades. In this chapter, background and literature on surface reconstruction from sample points are reviewed. There are various methods to construct surfaces. Skeletal modeling is a natural way to model objects, and is used mainly with implicit surfaces as a modeling tool. The convolution surface is a type of implicit surface. Background and literature on convolution surfaces are reviewed.

2.1 SHAPE MODELING FROM SKELETONS

As mentioned in Chapter 1, skeleton primitives include points, line segments, curves, etc. Points are widely studied as skeleton primitives. Based on the type of surface generated, the methods can be categorized as parametric representations, simplicial representations and implicit representations. Each of these three categories is discussed below.

2.1.1 Parametric Representation

Parametric curves and surfaces have traditionally been favorite modeling primitives in CAD systems. A rich literature exists on techniques for free-form modeling with these primitives. Parametric functions are usually defined in terms of control points. These methods represents surfaces as a parametric surface patches (e.g., Bézier, B-spline, Non-Uniform Rational B-splines (NURBS), etc.) described by parametric equations. For example, the equation of a Bézier surface defined by m+1 rows and n+1 columns of control points ($\vec{p}_{i,j}$) is:

$$P(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} B_{m,i}(u) B_{n,j}(v) \vec{p}_{i,j} \ , \tag{2–1}$$

where $B_{m,i}(u)$ and $B_{n,j}(v)$ are the i-th and j-th Bézier basis functions of degree of m and n in the u and v parametric directions, respectively. The basis functions are defined as follows:

$$\begin{aligned} B_{m,i}(u) &= \frac{m!}{i!(m-i)!} u^i (1-u)^{m-i}, \\ B_{n,j}(v) &= \frac{n!}{j!(n-j)!} v^j (1-v)^{n-j}. \end{aligned} \tag{2–2}$$

Parametric surfaces provide a natural method to control shape by manipulating control points.

Gelston and Dutta [GD95] use parametric surfaces to convert a geometric skeleton to a boundary representation. The skeleton of any 3D object consists of curves and surfaces. Skeletal curves and skeletal surfaces, i.e. spline curves and Coons patch surfaces, are used in their work. They then reconstruct the boundary surfaces based on a polynomial radius function and use a cubic spline curve as its spine curve. The surfaces are not always smooth, and constraints must be enforced in order to maintain G^1 continuity.

Goshtasby [Gosh03] presents a surface modeling method called the plus surface that adds a control plane to the parametric surface. The Bézier plus surface is defined in terms of a Bézier surface and an adjustment to it, defined by the control-plane normals. The Bézier+ surface has the form:

$$P(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} B_{m,i}(u)B_{n,j}(v)\vec{p}_{i,j} + \quad s\sum_{i=0}^{m}\sum_{j=0}^{n}[p^{u}(u_{i,j},v_{i,j})(u-u_{i,j}) + p^{v}(u_{i,j},v_{i,j})(v-v_{i,j})]B_{m,i}(u)B_{n,j}(v), \tag{2–3}$$

where s represents the magnitude of all control-plane normals and $p^{u}(u_{i,j},v_{i,j})$, $p^{v}(u_{i,j},v_{i,j})$ are the gradients of the ijth control plane with respect to u and v. The initial values of $p^{u}(u_{i,j},v_{i,j})$, $p^{v}(u_{i,j},v_{i,j})$ are estimated from the control points, and they can modified by the user interactively.

2.1.2 Simplicial Representations

In this representation the surface is modeled by a group of simple entities including points, edges and triangles. Unorganized point sets are the most studied examples of this type of representation. By skeletal-based surface model, we mean that the skeleton approximates the shape of the final complex object. An organized point set can also represent the shape of an object. We consider them as skeletal elements.

The first and most widely known reconstruction algorithm is the work of Hoppe et al. [HDDMS92] This algorithm estimates a tangent plane at each sample point using the k nearest neighbors, and uses the distance to the plane of the closest sample point as a signed distance function. The zero set of this function is then contoured by a continuous piecewise-linear surface using the marching cubes algorithm [HDDMS92].

One popular approach in this category is to reconstruct a triangulated surface using Delaunay triangulations[6] and Voronoi diagrams[7]. The reconstructed surface is

[6] A Delaunay triangulation for a set P of n points in the plane is a triangulation $DT(P)$ such that no point in P is inside the circumcircle of any triangle in $DT(P)$.

[7] Let P be a set of n distinct points (sites) in the plane. The Voronoi diagram of P is the subdivision of the plane into n cells, one for each site.

typically a subset of the faces of the Delaunay triangulations. Much work has been done in this area, and a review can be found in [Edel00, Edel98].

The alpha shape introduced by Edelsbrunner et al. [Edel83] is a subcomplex of the Delaunay triangulation. The alpha shape of a finite point set *S* is a polytope that is uniquely determined by *S* and a parameter α. The alpha shape expresses the intuitive notion of the "shape" of *S*, and α controls the level of detail reflected by the polytope. A Delaunay simplex (edge, face, etc.) belongs to the alpha shape of *S* if its circumsphere has radius of at most α. In their later work, Edelsbrunner et al. [Edel92] have extended this notion to *weighted alpha shapes*, in which the data points can be assigned scalar weights to cope with non-uniform sampling. Edelsbrunner and Mucke also extend the alpha shape from 2D geometry to 3D [EM94]. However, if the underlying surface is not sufficiently smooth, the alpha shape will in general have finite thickness. Another drawback of using the alpha shape for surface construction is that the optimal value of α depends on the sampling density, which often varies over different parts of the surface.

Rusak et. al. [RHVKJ00] develop a shape conceptualization modeling system based on particles. In their model, the alpha shape is employed as a geometry modeling tool to construct their conceptual model.

Bajaj et. al. [BBX97] construct the surface based on alpha shapes, and then a piecewise-algebraic surface fitting is applied. In their later work [BBCS99], a mesh reduction technique is used to simplify the mesh while maintaining the topology of the model.

Adamy et al. [AGJ00] introduce a method called the umbrella to model the unorganized points, which is based on a Delaunay complex. For a data set $S \in R^d$, an algorithm based on the "umbrella condition" is implemented to identify incorrect topology of surfaces. A complex is called a surface complex if the neighborhood of every

point $\vec{p}$ is homeomorphic to a $(d-1)$ dimensional closed disk, and $\vec{p}$ is not a point of its borders. Such a neighborhood is called an *umbrella*. In their algorithm, the Delaunay triangulation is computed, and an interval is assigned to each complex, which indicates the possibility of the nearby sample points. After that the optimal umbrella is selected at each vertex, the surfaces are constructed as the union of all umbrellas.

Some researchers focus on constructing the medial axis from unorganized points, and then reconstruct the surface. A rich literature can be found from the survey conducted by Attli et al. [ABE07] The *medial axis,* also called *medial axis transform (MAT),* is a shape model that represents an object by the set of maximal balls that are completely contained within the object. The medial axis (often called the medial *surface* in 3D) consists of the centers of the balls, and can be intuitively thought of as the *skeleton* of the object. In a sense, the MAT method is the reverse to our skeletal based modeling method, since we construct the surface from the skeleton, while the MAT method extracts the skeleton from the surfaces.

Amenta et al. [ACK01] reconstruct the surface using the so-called power crust algorithm. The power crust is a construction that samples a three-dimensional object and produces a surface mesh and an approximate medial axis. The algorithm first computes the poles from the samples using a Voronoi diagram, and the MAT is formed by these poles. They, then, apply an inverse transform to the MAT by using a power diagram (a weighted Voronoi diagram) to produce a piecewise-linear surface approximation.

Tam and Heidrich [TH03] present an algorithm for simplifying the shape of a 3D object by manipulating the MAT. They use Amenta's algorithm to generate the MAT, and then the MAT is simplified while conserving its topology. A peeling approach is used in which the outer layers of components are removed over a number of iterations. At the beginning of each iteration, the medial axis is decomposed into parts. Then a

significance value is assigned to each part that is a candidate for removal. Each part with a significance value in a given range can be deleted while maintaining the topological consistency. New surfaces can be reconstructed based on the filtered MAT.

Dey and Zhao [DZ02] present a method for computing approximations of the MAT from the Voronoi diagram of a set of sample points. These filters are also used to eliminate noisy components. They simplify the medial axis by simplifying the mesh while preserving the shape.

2.1.3 Implicit Representations

Implicit surface methods try to find a smooth function that passes through all positions where the implicit function evaluates to a specified value. An implicit surface S can be defined as a set of points $\vec{p} = (x, y, z)$ (where *x, y,* and *z* are the coordinates of the point) that satisfy $S = \vec{p} \mid \{f(x, y, z) = 0\}$. It is convenient to think of an implicit surface as an iso-surface formed in a scalar field $f(x, y, z)$ at a certain threshold T:

$$S = \{(x, y, z) \mid f(x, y, z) = T\}. \tag{2–4}$$

The field function f is often presented as:

$$f(x, y, z) = \sum_{i=1}^{n} f_i(x, y, z). \tag{2–5}$$

The summation is performed over all constituent functions, $f_i(x, y, z)$, each of which defines a density distribution in 3D-space.

An overview of implicit surfaces is given by Opalach and Maddock [OM95]. Implicit representations provide analytical formulas to generate the surfaces, with the advantage of producing smooth surfaces [OM95].

The blobby model [Blin82] was developed to represent electron density maps in molecular structures, but evolved into a general modeling tool. The blobby model is given by

$$h(r) = a \bullet e^{-br^2} . \tag{2–6}$$

Parameters b and a control the height and the width of the distribution. The function should be recognized as a Gaussian curve centered at the origin (actually a half Gaussian as $r \geq 0$). This approach is used to blend primitives together into smooth, continuous shapes.

Soft objects [NHKSO85] were developed as piecewise polynomial approximations to the Gaussian blending summation functions in the blobby model. This field function is basically the first few terms in a series expansion of an exponential truncated to restrict the range of influence:

$$h(r) = \begin{cases} a\left(1 - \dfrac{4r^6}{9b^6} + \dfrac{17r^4}{9b^4} - \dfrac{22r^2}{9b^2}\right), & r \leq b, \\ 0, & r > b, \end{cases} \tag{2–7}$$

where a is the scale factor for the function, and b is the distance after which each control primitive has no influence.

Metaballs [WMW86] are another piecewise polynomial approximation to Blinn's exponential functions. They use piecewise quadratic functions, and are very common in commercial modeling packages. Metaballs are represented as:

$$h(r) = \begin{cases} a(1 - 3r^2 / b^2), & 0 \leq r \leq b/3, \\ \dfrac{3a}{2}(1 - r/b)^2, & b/3 \leq r \leq b, \\ 0, & b \leq r, \end{cases} \tag{2–8}$$

where a is a scaling factor, and b is the maximum distance from which a control primitive contributes to the field.

Akleman and Chen [AC98] present a methodology to reconstruct objects from computed tomography (CT) scans using skeletal implicit surfaces. The skeleton is constructed from 2D Voronoi skeleton of each CT scan section. However, no proper method of surface construction is reported in this paper.

Crespin et al. [CBS96] present a method in which several primitives are used with an iso-surface model for geometric modeling by implicit surfaces. The basic idea of this work is to adapt the principle of sweep objects that exist in parametric modeling to implicit surfaces. A primitive P is defined as a single source point with one or several additional curves that control the shape of the field function. One primitive generates a set of concentric star-shaped iso-surfaces depending on the threshold value. The maximum extent of the field is called the front surface. The ratio between the distance from a given point M to the primitive P and the distance from P to the intersection of $\overrightarrow{PM}$ and the front surface is used to compute the potential function so that the final value of the field at M can be obtained.

Farley et al. [Far97] developed a relatively complex solution for generating smooth branches between skeleton curves from scattered data points. A branch is defined as the set of objects for which the geometric skeleton can be represented as a graph of interconnected curves without any surface elements. After the MAT (skeleton) is constructed from the scattered data points, the skeleton is converted to skeletal elements that can be used in iso-surface generation, that is to split the skeleton into a set of polylines (or branches), connected by the joints. If several branches have approximately the same size, a bulge appears near the junction. To solve the problem, they use the Bézier patch as a skeletal element to replace the joint branch to remove the bulge.

However, the method only works for modeling branches between three curves, which is quite limiting.

Turk and O'Brien [TO02] describe a method to model geometry with interpolating implicit surfaces. These surfaces are smooth and interpolate a set of given points, so users can easily grasp and reshape these surfaces with no thought to the underlying parameters of the model, which is an advantage over other implicit surface methods. They use radial basis functions[8] in a 3D domain to specify the location of an implicit surface. Radial basis functions are radially symmetric about a single point, or center. The surface is then interpolated as an approximation function, which is represented as a sum of *N* radial basis functions, each associated with a different location (the points), and weighted by an appropriate coefficient. Good results are reported from this paper. However, the implicit surfaces in their work are described by specifying locations in 3D through which the surface should pass, and also identifying locations that are interior or exterior to the surface. This approach is not applicable to our problem, as the surface location is unknown for our case.

Mari et al. [MS01, MAR00] present a method to reconstruct a surface from sample points. They build a deformable model of an implicit surface defined using a potential function and a skeleton. The medial axis is computed first, and the surface is then generated based on a union of balls. Local control of the surfaces is provided by dividing the data points into partitions related to seeds (the skeletal element). The constructed surfaces produced by this method are not necessarily smooth.

Yoo et al [YMS01] reconstruct the surface from unstructured samples from 2D image slices using variational implicit surfaces. Each individual 2D image slice is

[8] A radial basis function (RBF) is a real-valued function whose value depends only on the distance from the origin, so that $\phi(x) = \phi(\|x\|)$, or from another point *c*, called a center, so that $\phi(x) = \phi(\|(x-c)\|)$. Any function φ that satisfies the property $\varphi(x)=\varphi(||x||)$ is a radial function.

segmented, and the aggregate contours from the slices are processed to form a variational implicit surface.

2.2 CONVOLUTION SURFACES

Convolution surfaces utilize a concept that is well known in signal processing, namely the modification of a signal by a filter [BS91]. In this application, the geometrical information, i.e., the skeleton, is filtered by a kernel function that allows high frequencies to gently drop off in the neighborhood of the skeleton. The resulting field around the skeleton is the field function. The convolution surfaces are then generated by applying different threshold values. Most implicit models, such as blobby models [Blin82], metaballs [NHKSO85] and soft objects [WMW86] use points as input primitives. Convolution surfaces overcome this limit by providing a rich set skeletal primitives to model the object.

2.2.1 Background

A convolution surface is an implicit surface based on a field function $f(\vec{p})$ obtained by the convolution of some kernel function $h(\vec{p})$ and a geometry function $g(\vec{p})$:

$$f(\vec{p}) = g(\vec{p}) \otimes h(\vec{p}) = \int_{R^3} g(\vec{r}) h(\vec{p} - \vec{r}) d\vec{r} \,, \tag{2–9}$$

where $\vec{r}$ is an arbitrary point of interest. The geometry function $g(\vec{p})$ gives the spatial distribution of potential sources. The kernel function $h(\vec{p})$ describes the decay of a potential function produced by each point on the object.

The convolution surface is normally represented as:

$$S = \{\vec{p} \mid T - f(\vec{p}) = 0\}, \tag{2–10}$$

where T is the threshold value. An example of a convolution surface and its components are shown in Figure 2–1. The geometry function is a line segment along the x axis.

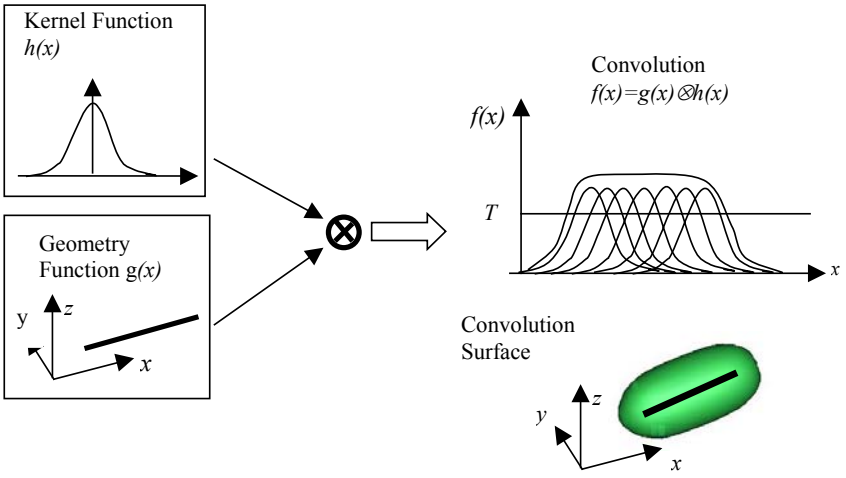


Figure 2–1. Convolution surface and its components [Sher99].

2.2.1.1 Kernel Function

Many decay functions can be used as the kernel functions, including blobby, metaball, and soft object, etc. Shersyuk [Sher99] presents a new kernel, the Cauchy kernel function, which is given by:

$$h(r) = \frac{1}{(1+s^2r^2)^2}, \quad r > 0, \tag{2–11}$$

where r is the distance from the point $\vec{p}$, and coefficient s controls the width of the kernel. A graph of the Cauchy kernel with s = 0.85 is shown in Figure 2–2. Shersyuk compares several kernel functions. The Cauchy kernel function gives analytic solutions to

most skeleton primitives. In this research, the Cauchy kernel function is used to model our convolution surfaces.

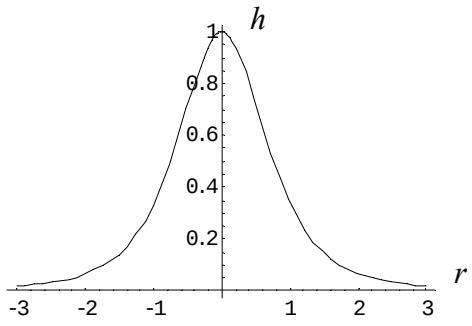


Figure 2–2. Cauchy kernel.

2.2.1.2 Field Function

The field function is the integral of equation 2–9. Different geometric primitives have different field functions. The field functions for a number of primitives have been derived. As an example, this section describes the analytic solution for the convolution surface for a line segment, which is shown in Figure 2–3. Appendix A presents field functions for other primitives including the sphere, line segment, arc, triangle, and plane.

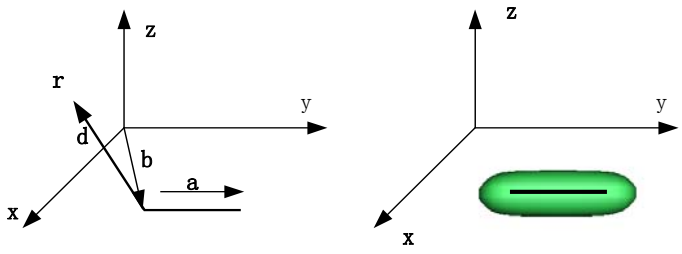


Figure 2–3. Example convolution surfaces of a line segment [Sher99].

A line segment of length l is defined as:

$$\vec{p}(t) = \vec{b} + t\vec{a},\ 0 \le t \le l, \tag{2–12}$$

where $\vec{b}$ is the base point and $\vec{a}$ is the unit direction vector. The squared distance from an arbitrary point $\vec{r}$ and a point on the line segment is:

$$r^2(t) = d^2 + t^2 + 2t\vec{d} \bullet \vec{a}, \tag{2–13}$$

where $d = \left\|\vec{d}\right\|$ is the magnitude of a vector from the segment base to $\vec{r}$: $\vec{d} = \vec{r} - \vec{b}$.

After integration, the field function for line segments is:

$$\begin{aligned} f_{line}(\vec{r}) &= \int_0^l \frac{dt}{(1 + s^2 r^2(t))^2} \\ &= \frac{h}{2p^2(p^2 + s^2h^2)} + \frac{l-h}{2p^2q^2} + \frac{1}{2sp^3}\left[\operatorname{atan}\left(\frac{sh}{p}\right) + \operatorname{atan}\left(\frac{s(l-h)}{p}\right)\right]. \end{aligned} \tag{2–14}$$

where $h = \vec{d} \bullet \vec{a}$, and p and q are distance terms:

$$\begin{aligned} p^2 &= 1 + s^2(d^2 - h^2), \\ q^2 &= 1 + s^2(d^2 + l^2 - 2lh). \end{aligned} \tag{2–15}$$

2.2.1.3 Properties of Convolution Surfaces

Superposition: If a single skeletal primitive is partitioned into two parts, the two abutting primitives will yield the same surface as the single more complex primitive, which is their union [BS91]:

$$h(\vec{p}) \otimes (g_1(\vec{p}) + g_2(\vec{p})) = (h(\vec{p}) \otimes g_1(\vec{p})) + (h(\vec{p}) \otimes g_2(\vec{p})), \tag{2–16}$$

where $h(\vec{p})$ is the kernel function, and $g_1(\vec{p})$ and $g_2(\vec{p})$ are the two geometry functions. An example is shown in Figure 2–4 with two line segments. This property allows us to separate the skeleton into individual parts. We can choose to analyze only those parts that cause topology changes, instead of analyzing all of them.

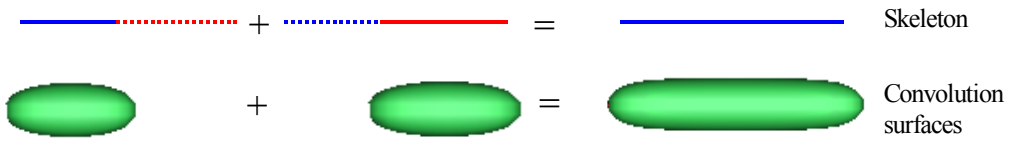


Figure 2–4. Superposition property of convolution surface.

Region of influence: An implicit surface can be characterized by its region of influence. A region of influence is the region in 3D space where the field function *f* of the primitives has non-zero values [Sher98]. For a convolution surface, the region of influence depends on the geometry of the primitive and the properties of the kernel function. An example is shown in Figure 2–5. Two spheres are used to demonstrate the region of influence. The dashed line boundary with radius *D* shows the region of influence. The radius of each convolution surface is *R*. Figure 2–5(b) shows the convolution surfaces. An important property of the region of influence is that convolution surfaces always fall within the boundary of these regions. This localizes the search for critical points.

a) stand alone objects with their Regions of influence

b) convolution surface

Figure 2–5. Region of influence.

2.2.2 Previous Work on Convolution Surfaces

Bloothmethal and Shoemake [BS91] introduce convolution surfaces into shape modeling with the ability to model the shape from line segments, polygons etc. Bloothmethal [BS91, Bloo95] convolves polygon-based skeletal primitives with a 3D Gauss kernel to generate the convolution surface. In their later work [Bloo97], they overcome some problems, such as bulges and curvature discontinuity where there is a field distribution of non-convex primitives. However, for higher dimensional skeletal elements, the Gauss kernel does not provide an analytical formula for constructing the convolution surface.

McCormack and Sherstyuk [SM98] introduce a new convolution kernel function called the "Cauchy" kernel. With the Cauchy kernel function, analytic formulas can be found for many geometric primitives, including point, line segments, triangles, curves, and planes. Blobby models, soft objects and metaballs can also be used as convolution kernels to generate convolution surfaces. However, these polynomial kernels cannot be convolved analytically with primitives whose dimensions are larger than the kernel's width. In his later work [Sher99], a complete design environment is described. The skeleton, region of influence, and blobbiness of all skeletal elements completely define an implicit surface. The region of influence is the region in 3D space where the field

function of the primitive has non-zero values, and the blobbiness parameter associated with the skeleton and passed to a kernel controls blending of the object's part [Sher99].

Jin et al. [JTFP01] model the shape by a convolution surface with polynomial weight distribution. They derived the analytic formula for line segment skeletons with non-uniform weights based on the Cauchy kernel function. The weights are modeled by polynomial functions. In their work, only line segments are used as skeletal elements.

Ma et al. [MJFP01] extend the geometric primitives to planar NURBS curves. In their work, they do not calculate the convolution surface integral for the curve element directly. Instead, they sample the skeleton curve base, construct and interpolate the bi-arc spline, and reduce the number of arcs within a prescribed tolerance. Finally, the potential functions of the arc spline primitives are summed. An analytical formula was not generated for the planar NURBS curve skeletal elements.

Zhao and Rockwood [ZR94] applied the concept of convolution surfaces to the *N*-sided patch. The *N*-sided patch is defined by *N* vertices and a tangent plane for each vertex. Each of its four-sided sub-patches arise naturally from a convolution surface, which interpolates one of the "predefined" vertices and its corresponding tangent plane. The convolution kernel is a bi-cubic piecewise Bézier patch that is convolved with bi-quadratic Bézier patches as simple geometric primitives to form the surface. In their method, a convolution surface patch is generated to blend the vertices. The method requires enforcement of constraints to provide G^1 continuity along common boundaries of the patches.

Cani and Hornus [CH01] introduce a new convolution kernel and extend the skeletal elements to subdivision curves. They also try to avoid unwanted blending. However, when two curves are placed very close, their method does not work. In their later work, they try to overcome this problem. Angelidis et al. [AJC02] introduce *local*

convolution in contact situations. This concept defines a restricted skeleton range for computing the field value at a certain point in space. A parameter is defined on each curve of the skeleton. This prevents blending even between small folds of the skeleton. In later work, a new kernel is presented to allow exact convolution with a varying radius [HAC03]. With the new kernel the field function preserves surface smoothness when different radii are specified along a poly-line. However, the skeletons studied in their work are restricted to graphs of branching curves.

Oeltze and Preim [OP04] present a method for visualizing anatomic tree structures, such as vasculature and bronchial trees based on clinical computed tomography (CT) or magnetic resonance imaging (MRI) data, with convolution surfaces. This method uniformly handles anatomic trees with different types of branches and produces smooth surfaces. They propose a way to model the surface that exactly converges to a given radius. Others [Sher99, JCLP01] do not consider the given radius at all. The radius is also related to the threshold value, but was not mentioned in this research.

2.2.3 Unwanted Blending

As mentioned before, unwanted blending in modeling with implicit surfaces causes problems. For example, for modeling implicitly defined characters, such as in Figure 2–6, the arms should blend with the shoulders (as shown on the left), but not with other parts of the body (as shown on the right) [AJC02]. In order to use implicit surfaces to model the shape, the system must provide blending control. Techniques for avoiding such unwanted blending have been developed.

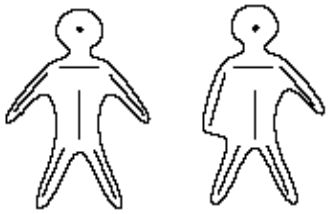


Figure 2–6. Unwanted blending [AJC02].

Cani-Gascuel [Cani93] warps the field function to simulate precise contact instead of unwanted blending. Kacic-Alesic and Wyvill [KW91] solve the unwanted blending with two methods. One is to group skeletons of objects that should not blend together. The group is then treated as a single skeleton, and the distance to the skeleton is the minimum distance to any of the elements of the group. Another solution is to use different blending functions for different skeletons.

Guy and Wyvill [GW95] formulate a solution to the unwanted blending problem that involves building a blending graph that describes the relationship between primitives. Primitives that the modeler desires to blend with each other form a *blending group*. However, the result is not always a smooth blend.

Angelidis et al. [AJC02] attempt to control blending using local convolution with restricted skeleton elements. A weight factor is attached to each segment of the skeleton. The total field value near a branching point is then defined by summing these contributions. To calculate the weight factor, sampling points are also needed. One problem with this method is that sample points from the surface are difficult to compute.

2.3 SUMMARY

From the above review, we can see existing limitations to skeletal modeling:

- The surface generated by a simplicial surface is smooth only when the density of the point sources is large. However, in conceptual design we want to use as few

geometric primitives as possible (or limited by lack of detailed information) to represent parts.

- Both parametric and simplicial surfaces only consider point sources as input. In conceptual design, users should have the flexibility to use various skeletal elements to construct the skeleton.
- Implicit surfaces have the ability to generate smooth surfaces, however, not all implicit surface methods can handle geometric primitives other than points.
- Convolution surfaces have the property of generating smooth surfaces and provide a rich set of skeletal primitives, i.e., points, line segments, triangle elements, planes, etc. However, surface topology can change arbitrarily. In engineering design the surface generated from a skeleton must be consistent with the topology of the skeleton. Topology analysis is necessary. However, most modeling systems do not consider topology changes.

In this research, Cauchy based convolution surfaces are selected to generate smooth surfaces for our skeleton based modeling system. However, the surface topology can change arbitrarily. The topology of the convolution surfaces should be controlled so that it is consistent with the topology of the skeleton. The background and literature about surface topology is reviewed in next chapter.

Chapter 3 Background and Literature on Surface Topology

Topology analysis of constructed surfaces has attracted many researchers in the computer graphics and computational geometry areas. By a *surface* we mean a "compact, connected, orientable two-dimensional manifold, possibly with a boundary, embedded in $\Re^3$ [O'Nel66]. In this chapter, the background and literature of topology are reviewed. One of the key tools for analyzing the topology over a smooth surface is Morse theory. This theory establishes the relationship between the critical points (CPs) and the topology changes of the surface. The fundamentals of Morse theory are introduced, and research related to Morse theory is discussed. Matching topology of an initial mesh configuration to a new configuration is also discussed in this chapter.

3.1 TOPOLOGY

One of the most important developments in mathematics in the twentieth century has been the development of topology as an independent field of study and the subsequent systematic application of topological ideas to other fields of mathematics. As noted in Chapter 1, topology is an abstraction of certain geometrical ideas such as "continuity" and "closeness" [FG01]. Topology is the mathematical study of the properties of surfaces that are preserved through deformations other than tearing [Weis07]. In general, the topology of a surface refers to the connectedness of the shape.

The study of topology began with the investigation of certain questions in geometry. It has been studied from various mathematical angles and applied to various types of data ranging from point sets and curves, to manifolds. Regardless of the methods

used to study topology the fundamental thrust is always the same, the study of the spaces independent of geometric metrics.

3.1.1 Topology Invariants

Two kinds of topological invariants are used to measure the topology of the surfaces: the Euler characteristic and Betti numbers. Each of these is described in this section. We use both for our topological matching approach, as described below.

3.1.1.1 Euler Characteristic

One topologically invariant property of a surface is defined as the largest number of nonintersecting simple closed curves that can be drawn on the surface without separating it [Weis07]. Roughly speaking, this is the number of holes in a surface, which is called the *genus*, *g*. The genus of a surface, also called the geometric genus, is related to the Euler characteristic χ, defined as

$$\chi = V - E + F, \tag{3–1}$$

where *V, E, F* are the numbers of vertices, edges, and faces of the surface, respectively.

Once we know the Euler characteristic χ, we can calculate the genus *g* as:

$$\chi = 2 - 2g. \tag{3–2}$$

3.1.1.2 Betti Numbers

Another set of invariants used to describe surface topology properties is the Betti numbers, consisting of three individual numbers: β_0, β_1 and β_2. There are simple physical interpretations of the Betti numbers. Loosely defined, β_0 is the number of gaps that separate components, that is, the number of connected components; β_1 is the number

of tunnels that pass through the shape; and β_2 is the number of voids that are components of the complement space inaccessible from the outside [DES99].

The relationship between the Euler characteristic χ and the Betti numbers is:

$$\chi = \beta_0 - \beta_1 + \beta_2 . \qquad (3\text{–}3)$$

Recall that the topology of a surface refers to the connectedness of the shape, which includes the number of disconnected components, the number of tunnels and the number of voids in each component. As described above, the Euler characteristic only has one value, which represents the number of holes in the object. When comparing the topology of convolution surfaces, we are not only interested in the number of holes, but also in the number of voids and the number of components. Betti numbers provide rich topological information about surfaces; hence in this research, Betti numbers are used to characterize the topology of skeletons as well as convolution surfaces.

3.1.2 Characterizing Topology

One of the algorithms proposed in this research is a method to identify the topology of the convolution surface. We first consider other methods reported for identifying the topology of a surface. Dey et al. [DES99] review a number of examples of computational topology, which focus on topology analysis of shapes in general, as opposed to surfaces generated from skeletons.

A combinational approach for computing Betti numbers is described in [DE95]. The basic idea of the algorithm is to build up a simplicial complex by adding one simplex at a time. The Betti numbers are calculated based on the connectivity between the triangle elements.

Edelsbrunner et al. [ELZ02] describe a method for computing topology persistence of simplicial complexes, in particular, alpha complexes. They use Betti

numbers to measure the topological complexity of a point set in $\mathbf{R}^3$. In this paper, the filter, which is the ordering of the simplices, plays an important role in the algorithm. The filter is assumed to be "alive" and associated with the growth of the complex. The Betti numbers are then calculated based on the growth of the filters.

El-Sana and Varshney [ESV98] address the problem of controlled topology simplification for polygonal models. Their approach identifies the genus by rolling a sphere of small radius over the object and filling tunnels that are not accessible. They use heuristics to detect holes and other cavities commonly found in CAD models.

Szymczak and Vanderhyde [SV03] introduce an algorithm for extracting topologically simple iso-surfaces from regularly sampled volume data. The algorithm provides the user with precise control over the genus of the extracted iso-surface. Such control is particularly desirable when the true topology of the iso-surface to be extracted is known *a priori*. The volume data iso-surfaces are generated by a variant of the marching cubes algorithm.

Wood et al. [WHDS02] present an algorithm to simplify the topology of an iso-surface by introducing small local changes to the volume. They explicitly analyze the topology of the input iso-surface based on the Reeb Graph[9]. For each small handle, an associated non-separating loop in the iso-surface is found and the values of voxels located along its spanning disk are changed so that the handle disappears.

Guskov and Wood [GW01] introduce an approach that removes unnecessary nontrivial topology from meshes. Using a local wave front traversal, the local topologies of the mesh are discovered and features such as tunnels are identified.

[9] A Reeb graph is a compressed representation of the surface components; it represents the connectivity of the level set.

3.2 MORSE THEORY

Morse theory is the tool used in this research to study the topology of spaces. Morse theory gives a direct way of analyzing the topology of a manifold by studying the differential function on that manifold. The critical points of the surface are keys to predicting the topology of the manifold. Morse theory articulates the relationships between the CPs of a smooth function defined on a smooth manifold to the global topology of the manifold [Miln63]. The fundamentals of Morse theory are presented in this section. For a more detailed treatment, refer to [Miln63, Hart98]. To better understand Morse theory, we first define a CP. In this research, a 2-manifold is analyzed in $\Re^3$ Euclidean space.

Definition 1: *Critical point (CP).* Let f be a smooth function with C^2 continuity with local coordinates (x, y, z) on a manifold M. A point $\vec{p} \in M$ is a CP *iff* its derivatives with respect to a local coordinate system on M vanish. That is:

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \right) = 0 . \tag{3–4}$$

Definition 2 *Critical value*: Let $\vec{p}$ be a CP of *f*. The real number $f(\vec{p})$ is called a critical value of f.

Definition 3 *Morse point:* Morse theory focuses only on non-degenerate CPs. Such points, also called Morse points, are CPs where the Hessian is non-zero. That is:

$$H(f) = \begin{Vmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\ \frac{\partial^2 f}{\partial x \partial z} & \frac{\partial^2 f}{\partial y \partial z} & \frac{\partial^2 f}{\partial z^2} \end{Vmatrix} \neq 0 . \quad (3\text{–}5)$$

Since $\partial^2 f / \partial x \partial y = \partial^2 f / \partial y \partial x$, the matrix *H(f)* is symmetric with real eigenvalues. Let $\lambda_1 \leq \lambda_2 \leq \lambda_3$ be the eigenvalues of *H(f)*. If any of the eigenvalues is zero, then the CP is degenerate. Otherwise it is called non-degenerate.

Definition 4 *Index***:** The *index* of a CP is the number of negative eigenvalues of *H(f)*. For a 2-manifold in $\Re^3$ Euclidean space, a CP of *index* 0 or 3, is called a minimum or maximum, respectively. There are two types of saddle points, which can be classified based on the number of negative eigenvalues of *H(f)*. If the number of negative eigenvalues of *H(f)* is 1, the CP is an index 1 CP, also called a saddle 1. An index 2 CP is also called saddle 2 CP, and has two negative eigenvalues of *H(f)*.

Definition 5 *Morse function:* A smooth function on a smooth manifold is called a Morse function.

Morse Lemma: It is possible to choose local coordinates (x, y, z) in a neighborhood of a non-degenerate CP $\vec{p} \in M$ so that the Morse function takes the form:

$$f(x, y, z) = T \pm x^2 \pm y^2 \pm z^2 , \quad (3\text{–}6)$$

at the CP (0,0,0), where T is the threshold value. The existence of these neighborhoods means that the CPs are isolated, i.e., they have neighborhoods that are free of other CPs.

Morse theory not only indicates when the topological type changes, but what kind of change takes place [Hart99]. Such changes are determined by the type of the CP as well as its value. The relationships between the CP types and the topology changes are listed in Table 3–1.

Table 3–1. Relationship between CP type and topology changes.

Sign of Eigenvalues ($\lambda_1 \le \lambda_2 \le \lambda_3$)			CP type	Topology Change
λ_1	λ_2	λ_3		
-	-	-	Index-3	Void formed or filled
-	-	+	Index-2	Loop formed or filled
-	+	+	Index-1	Objects connected or disconnected
+	+	+	Index-0	Object appears or disappears

3.2.1 Previous Work on Morse Theory

Morse theory has been used to reconstruct surfaces from cross sections [SKK91], to find surface-surface intersection curves [Cheng89], and to detect the topology of algebraic surfaces [BCL96], and implicit surfaces [SH98]. Bajaj and Schikore [BS96] solve a surface simplification problem by using Morse theory to simplify terrain data while ensuring that the terrain retains the structure of its "peaks and passes." The simplification does not allow the intersection or collapse of a vertex if it causes a change in the critical point structure of the terrain.

Other researchers have addressed the challenges of applying Morse theory to the discrete setting. Axen and Edelsbrunner [AE98] apply Morse theory to triangular meshes in the form of compact piecewise linear 2-manifolds, using a wave traversal technique. The wave traversal is similar to the height function. First, an initial mesh vertex is chosen

and assigned a height of zero. Then all of the vertices sharing an edge with that vertex are selected and assigned a height of one. The process continues until all of the vertices are assigned a distance from the initial vertex. Examining all vertices at a given distance as the distance increases yields a wave traversing through the triangular mesh.

Axen [Axen99] introduces algorithms to compute a discrete Morse function on a triangulated manifold. The Morse function is calculated based on the wavefront traversal method, and the critical points are found by examining the neighborhood of each vertex.

Wood et al. [WHDS04] focused on 2-manifold topology by detecting the topology in an area of interest, where the genus is one. However, only handles (tunnels) are considered in this work. Tunnels are only one type of topology characters; others should be considered as well.

3.2.2 Critical Points

The analysis of topology using critical points is not new to the field of geometric modeling. The work of Bajaj and Schikore [BS98] uses critical point analysis to constrain geometry simplification to be topology preserving. Boissonnat et al. [BCV03] use critical points to solve the problem of iso-surface meshing with topology guarantees based on Morse theory.

Critical points have been used to determine implicit surface topology during a "shrinkwrap" polygonization [BNV96]. A topology change occurs at a critical point. The distinction between a rupture and a hole is made on the basis of a local approximation of the function near the critical point as a quadratic polynomial.

Critical points of vector and tensor fields are used to delineate topologically distinct regions in the visualization of flow [HH91, DH94]. While the critical points are not used to determine changes in topology, the techniques used to detect critical points can be used to support Morse theory application.

3.2.3 Detecting Critical Points

Stander and Hart [SH98] use interval analysis to determine the search regions for individual critical points. A search over the entire domain is conducted based on interval analysis. After each subregion containing only one critical point is identified, Newton's method is used to determine the exact location of each critical point.

Wu and Malheiros [WM99] analyze the topology of implicit surfaces. The spatial coherence of the skeletal elements is used to predict the sites of the critical points. An incidence graph is then constructed to represent the topology of the generated surfaces. This method is restricted to point based implicit surfaces.

Siersma [Sier99] proves that Voronoi diagrams and Delaunay triangulation can be used to predict the saddle points for non-differentiable 2D surfaces. In his research, the original surfaces are simplified by triangulation. He proves that the Voronoi points are maximum points and the intersection of Voronoi edges and Delaunay triangle edges are the saddle points. However, the method only works with point primitives and is not suitable for other higher dimensional primitives.

Malcolm and Popelier [MP02] present a new algorithm for locating the critical points in general scalar fields for analyzing quantum chemical topology based on their previous work [Pope98, Pope96]. The gradient paths, which are paths of steepest ascent through a scalar field S, are used to describe the connectivity between different types of critical points. Several possible types of gradient paths are observed, each of which start swith one type of critical point, and endswith another type of critical point. The paths are then used to improve the seeding search algorithm, which provides a suitable set of starting points for the search for critical points. However, the method for finding the gradient path is not clear in this paper, and it only works with point primitives.

3.3 MATCHING TOPOLOGY

A variety of researchers focus on matching the topology of a given mesh to a new configuration [AC98, ACS00, Thom00, SH99, VGW94, GP00, NPK06]. Ning and Bloomenthal [NB93] review several polygonization algorithms, focusing on topological consistency and correctness. Topological correctness implies that the polygonal approximation is homeomorphic to the implicit surface. Topological consistency only requires that the polygonal approximation is consistent with itself. That is, the polygonal model contains no holes or dangling polygons. Some of the surveyed algorithms guarantee topological consistency, but none of the surveyed algorithms guarantee topological correctness for continuous data sets.

Stander and Hart [SH99] guarantee that the topology of the polygonization of the implicit surface agrees with the topology of the surface. The critical points are tracked to make sure the polygonization of the surfaces has the same topology as the implicit surfaces.

In Thompson's [Thom00] skeletal modeler, topology matching between the offset surface and its skeleton is ensured by forcing the offset surface to fall inside the union of balls. A similar method is applied in [MG01] to a union of balls model, guaranteeing explicit topological control due to the connectivity links between the points of the skeleton.

Gerstner and Pajarola [GP00] report a method that allows preservation and controlled topology simplification. They focus on the surface underlying the 3D volume data set itself. A coarser iso-surface approximation has the same genus as the highest resolution iso-surfaces. They extract the iso-surface directly from the 3D volume data. Some of the vertices of the tetrahedron are found to be critical points.

Nieda et al. [NPK06] present a method based on Morse theory to match the topology of surfaces during metamorphosis. They present an augmented Morse analysis algorithm to reduce the computing time for detecting critical points significantly. However, detrails of the algorithm for detecting critical points are not given.

Varadhan et al. [VKSM04] describe an approach to computing topology preserving iso-surfaces that arise in a variety of geometric processing applications. They also present a sufficient condition for sampling a distance field so that the reconstruction maintains the topology of the original iso-surfaces. However, the algorithm may not handle degenerate configurations in the input model.

Bischoff and Kobbelt [BK02] present an algorithm that is able to extract iso-surfaces with controlled topologies for medical applications. The idea is to find an initial estimate of the final surface that can be guaranteed to have a desired topology. A topology preserving growing scheme is then applied that morphs the initial estimate into the final result.

3.4 SUMMARY

Morse theory is a powerful tool for analyzing the topology of smooth surfaces and discrete data sets. Topological analysis of some types of implicit surfaces with Morse Theory is reported. However, most examples of topology analysis are based on point primitives. No topological analysis of convolution surfaces with higher dimensional skeletal elements is reported.

Several algorithms for searching for CPs of reconstructed surfaces are reported based on interval analysis, or guided by the properties of an underlying triangular mesh. Using the skeleton as a guide to search for CPs is reported by Wu and Malheiros; however, our method is different from theirs, as we partition the space to locate the

critical point. In addition, we focus on higher dimensional skeletal primitives, such as line segments and triangles, while their algorithm works with point primitives only.

A variety of researchers focus on matching the topology of a given mesh to a new configuration. However, no method is reported for matching the topology of the surfaces with the root skeleton.

In the next chapter, we will present a method to examine the topology of the constructed convolution surfaces to ensure topological agreement between the surfaces and the skeleton.

Chapter 4 Searching for Critical Points of the Convolution Surfaces

As described in Chapter 2, convolution surfaces have the advantages of generating smooth surfaces from the skeleton and providing a richer set of geometric primitives, which offers the designer more options for defining conceptual geometry. For these reasons, Cauchy based convolution surfaces were selected to generate smooth surfaces for our skeleton based modeling system. However, the nature of Cauchy convolution surfaces is such that the surface topology can change arbitrarily. We assert that the topology of the convolution surfaces should be controlled to be consist with the topology of the skeleton. Morse Theory, reviewed in Chapter 3, is a tool for analyzing the topology of smooth surfaces. The theory draws a relationship between the critical points (CPs) of the implicit surface function and topology changes of the surfaces. In order to use Morse theory to analyze the topology of the convolution surfaces, the CPs of the surfaces must be identified. In this chapter, we present an algorithm for locating the CPs of convolution surfaces used in a skeletal modeling system. A strategy is discussed that uses knowledge of the skeleton to guide the search for the CPs. The fundamentals of sequential quadratic programming (SQP) are introduced, and the implemented SQP algorithm for finding CPs is illustrated with two case studies.

4.1 SEARCHING SPACE

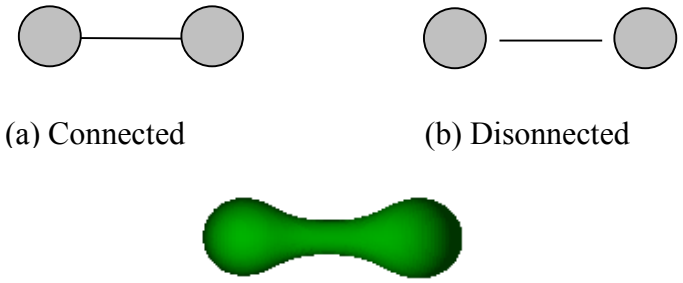
In order to find the CPs of the function, a search over the domain of the function must be performed. However, Cauchy function based convolution surfaces present problems for search algorithms. The natural search region is defined by the boundary of the skeleton. Because there is normally more than one CP associated with a skeleton, the

search algorithm must have the ability to locate all CPs. The basis of our approach is subdividing the search area (defined by the skeleton boundary) into multiple small regions, each containing a single CP. Stander and Hart [SH99] use interval analysis to identify search areas (polygonized implicit surfaces in their case) containing a single CP. Newton's method is then employed to find the CPs. The main objective of their work is to guarantee that the topology of the implicit surfaces and the topology of the polygonal approximation match. Wu and Malheiros [WM99] constructed an incidence graph to locate the search area. The graph contains loops and connection relationships among the primitives. With the identified CPs, the topology of the generated surfaces can be calculated. However, our problem, which is different from theirs, is to maintain the same topology between the convolution surfaces and the skeleton. We need to know not only the topology of the convolution surfaces, but the topology of the skeleton as well. Therefore we propose that the skeleton can be used as a guide to find the CPs for the convolution surfaces.

4.1.1 Interpreting the Designer's Intent

In our proposed conceptual geometric modeling tool, the designer uses a skeleton to specify the initial shape of the product. Smooth surfaces, convolution surfaces, are constructed based on the skeleton. To provide an effective modeling environment, certain assumptions must be made to enable topological consistency checking between the skeleton and the convolution surfaces of the final design. We propose that the skeleton not only represents the final shape of the design, it also represents the desired topology of the final design. We expect the generated convolution surfaces to have the exact same topology the skeleton. This assumption allows us to use the skeleton as a guide for finding the CPs for the convolution surfaces. Further, we assume that the designer intends for the skeleton to model a single object, which means all skeletal primitives must be

connected. The skeleton and resulting surfaces are considered to be one connected part. If the designer wants to model several parts in a single design, they must be modeled separately. Figure 4–1 illustrates why we make this assumption. Figure 4–1(a) shows the skeleton of a simple dumbbell, which is formed by two spheres connected by a beam. In contrast, the skeleton in Figure 4–1(b) consists of three different, disconnected parts. Both skeletons can generate the same final shape: the dumbbell shown in Figure 4–1(c). The topology of (a) is different from that of (b), and is consistent with the surface in (c). In (a) there is one single object, while in (b) there are three objects. We assume that the designer intends for three separate objects to result from the skeleton in Figure 4–1(b). Thus, to distinguish the two cases, we do not support case 2(b) in our modeler. The designer would have to model the three objects separately.



(a) Connected (b) Disonnected

(c) Final shape

Figure 4–1. Two different dumbbell skeletons and the resulting convolution surface.

4.1.2 Heuristics for Locating Critical Points of Convolution Surfaces

Convolution surfaces, in this research, are the level set of the field function, which is the sum of all geometric primitives of the skeleton with convolve with the Cauchy

kernel function. Locating the critical points of convolution surfaces requires determining the zeros of a non-linear function, i.e., the gradient of the surface equation. This has proven to be intractable in closed form and requires the use of numerical approaches. Numerical search methods are generally known to be sensitive to the locations of starting points, and success relies on identifying good starting points. We have developed a number of heuristics that guide the choices for starting points. These heuristics are based on observations about where the critical points for different skeletal primitives are located in different areas. In this section, we describe these heuristics.

- ***Locations of CPs of Point Primitives:***

  Index-0 CP: The index-0 CP is located near the center of the point primitive, although not exactly at the center, since the generated convolution surfaces are affected by other skeletal primitives as well.

  Index-1 CP: An index-1 CP indicates whether two objects are connected or not, so it is normally located between two points. Between two index-0 CPs, there is one index-1 CP.

  Index-2 CP: Index-2 CPs indicate loops. An index-2 CP appears in an area surrounded by index-1 CPs, which is normally located in the convex area bounded by the point primitives.

These observations are similar to Siersma's results [Sie99]. Siersma used the Voronoi diagram and Delaunay triangle edges to predict the locations of saddle points for non-differential surfaces. He considered the minimal distance function of a point in the plane to a set P of N points in the plane. He proved that the saddle point occurs exactly where a Delaunay edge cuts the corresponding Voronoi edge in its interior. Voronoi points are maxima and the set of points P are the minima. An example is illustrated in Figure 4–2. The figure shows two slightly different configurations consisting of three

points. The left configuration has a maximum point, 3 minima, and 3 saddles, while the right configuration only has two saddles and no maximum points. The corresponding convolution surfaces and the CPs are shown in Figure 4–3.

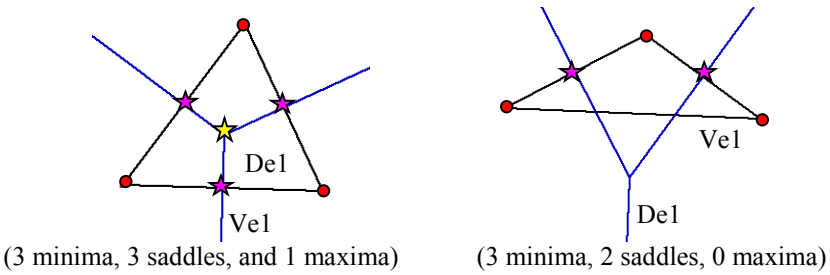


Figure 4–2. Three point configurations with Voronoi diagrams and Delaunay triangulations [Sier99].

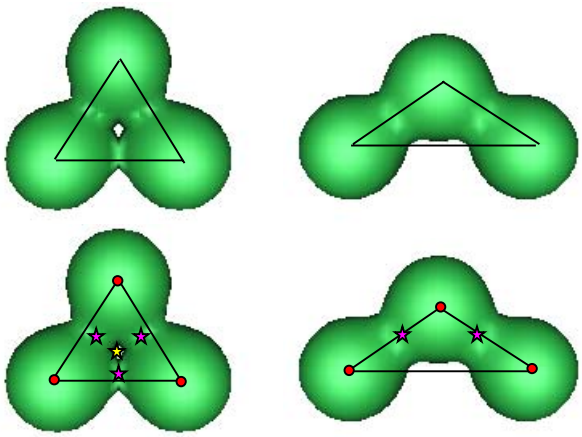


Figure 4–3. Corresponding convolution surfaces of the skeletons shown in Figure 4–2.

When searching for the CPs of the convolution surfaces, the actual CPs found are not those points shown in Figure 4–2. However, they are close. Convolution surfaces are the offset surfaces of the skeleton. Each geometry primitive has its own contribution to the surfaces. The influences from other geometric primitives are not considered when using Siersma's method searching for CPs, since triangulated surfaces are analyzed in his work. In addition, the algorithm Siersma [Sier99] reported only worked for skeletons with point primitives.

- ***Locations of CPs of Line Segments Primitives:***

The CPs of a convolution surface formed by a line segment are distinct from those of a point primitive. When an object is formed by a single line, the index-0 CPs are located at the center of the line segment. However, when several line segments form a complicated skeleton, the locations of the index-0 CPs move closer to the intersection points of pairs of line segments. The potential at the intersection point is higher than at other locations. Both geometry primitives have contributions to the surface in that region.

Index-0 CPs: Index-0 CPs appear near the intersection point of two line segments.

Index-1 CPs: Index-1 CPs appear near the center of line segments.

Index-2 CPs: Index-2 CPs are surrounded by index-1 CPs. They are normally located in the convex area whose boundary is the line segments.

- ***Locations of CPs of Triangle Primitives:***

The rules for triangle primitives are similar to those of point primitives. The index-0 CPs are located at triangle centers. There are two conditions under which two triangles connect with each other: edge-edge, or vertex-vertex. For the edge to edge connection case, an index-0 CP appears between the two triangle primitives. If the connection between two triangle elements is vertex-vertex, an index-1 CP appears between them. Index-2 CPs are located in regions surrounded by index-1 CPs. Similar to

the other two primitives, the index-2 CPs are normally located in the convex area bounded by triangle elements.

4.2 ALGORITHM

Based on our assumptions, every skeleton consists of a set of connected elements, and each element is connected to at least one other element. The final shape is always a single object. Thus, to save computational expense, we do not consider index-0 CPs. Currently, we consider 2D skeletons, and we only search for index-2 and index-1 CPs.

As described in section 4.1, there are different types of CPs, and more than one CP may appear for the skeleton based convolution surfaces. The general locations of the different types of CPs are observed, and they are directly related to the skeleton itself. In order to find all CPs, we need to search specific regions to locate each CP. For example, concave points in a skeleton might lead to the generation of convolution surfaces with unwanted loops. Therefore, the search areas should be partitioned to small regions so that each region only contains one CP. Based on our observations, the index-2 CPs are located in the convex regions of the skeleton. Partitioning the skeleton to convex regions will help us to locate the index-2 CPs.

4.2.1 Partitioning the Space

Several researches have reported algorithms to perform such partitioning [dBvKOS97, Gree83, HM83, LA04]. The computational geometry algorithms library (CGAL) [webCGAL] provides implementations of these algorithms with source code. The “2D convex hull” and “planar polygon partitioning” algorithms from the CGAL library are adopted here.

4.2.1.1 Convex Hull

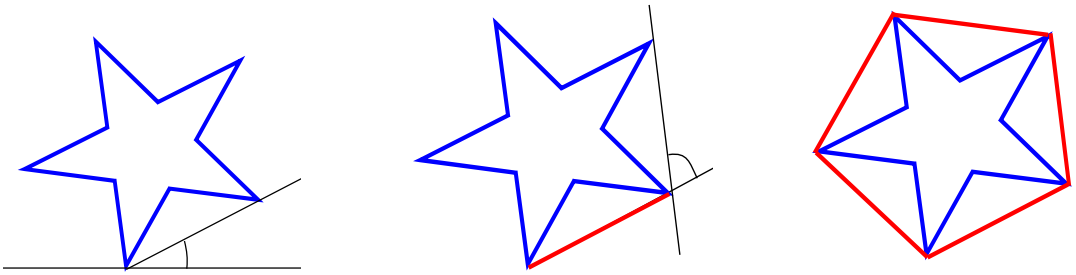
A set $S \in \mathbf{R}^2$ is convex if for any two points p and q in the set the line segment with endpoints p and q is wholely contained in S. The convex hull of a set S is the smallest convex set containing S. For a two-dimensional finite set the convex hull is a convex polygon.

There are many methods for computing the convex hull. CGAL provides implementations of several classical algorithms for computing the convex hull for a set of points in two dimensions. As an example, details of the wrapping algorithm is presented here.

Step 1. Find the lowest y-value point. Construct a line segment from that point to the point that is at the smallest angle from the horizontal.

Step 2. Find the next smallest angle relative to the line segment that was just found, and construct a line segment to the new point.

Step 3. Repeat step 2 until the original point is reached.



(a) step 1 (b) step 2 (c) step 3

Figure 4–4. An example of the wrapping algorithm.

Additional lines may be added when computing the convex hull. That is, the convex hull may produce additional lines that are not part of the skeleton. An example is shown in Figure 4–4. Assume that the skeleton is a star shape. Five additional lines (red lines) are added to form the convex hull. These five lines are not part of the skeleton, but they form the boundaries of the search regions, together with the skeleton.

4.2.1.2 Identify Polygons

The skeleton is formed by different primitives, and the connectivity between each skeletal element can be recorded. In this way, the number of loops in the skeleton is identified. To complete this task, the well known data structure halfedge [CGALweb, dBvKSO97] is applied. This representation records the connections between components. An example illustrating the concept of the halfedge is shoen in Figure 4–5.

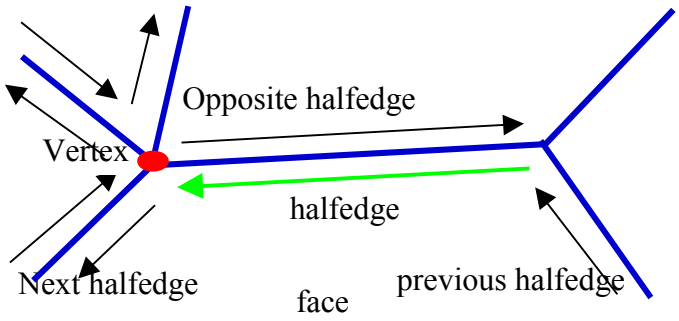


Figure 4–5. Halfedge data representation.

A halfedge data structure is an edge-centered data structure capable of maintaining incidence information of vertices, edges, and faces. Each edge is decomposed into two halfedges with opposite orientations. Each halfedge is associated

with exactly one vertex, edge, and face, defined as the vertex end of the directed edge, and face to the left of the edge, respectively.

The halfedge data structure consists of three types of objects: vertices, half edges, and faces. These objects primarily consist of pointers to other halfedge objects. Figure 4–6 shows example implementations of edge, vertex, and face data structures.

```
struct HE_edge {
        HE_vertex* vertex;
        HE_face*
        HE_edge* next; // next halfedge in the face on the left
        HE_edge* sym; // the other half edge for   the same edge
        }
```

(a) edge

```
struct HE_vertex {
        float x;
        float y;
        float z;
        HE_edge* halfedge;
        }
```

(b) vertex

```
struct HE_face {
         HE_edge* edge;
         }
```

(c) face

Figure 4–6. Halfedge data structure implementation.

The edges in the halfedge data structure contain a vertex (which is the end of the halfedge); an oppositely oriented adjacent halfedge; a face the halfedge is adjacent to;

and the next halfedge around the face. The vertices in the halfedge data structure store their *x*, *y*, and *z* position coordinates, as well as a pointer to exactly one of the helfedges that uses the vertex as its ending point. The face only needs to store a pointer to one of the halfedges that borders it.

With this data structure, the halfedges adjacent to a face can be found, since the halfedges around a face form a circular linked list, and the face structure stores a pointer to one of these halfedges. For our problem another property is added to the halfedge data structure: the type of the edge. This is an indicator that distinguishes whether the edge is a line segment, or whether it belongs to a triangle element, or whether it is an added line due to the convex decomposition of the skeleton.

4.2.1.3. Polygon Partition

Polygon partitioning is an important preprocessing step for many geometric algorithms, because most geometric problems are simpler and faster when applied to convex objects compared to non-convex objects. A partition of a polygon *P* is a set of polygons such that the interiors of the polygons do not intersect and the union of the polygons is equal to the interior of the original polygon *P*. Surveys of partitioning and covering polygon algorithms are given by Keil [Keil00, KS85].

CGAL provides the simple approximation algorithm of Hertel and Mehlhorn [HM83]. They describe a heuristic for convex decomposition using diagonals that is simple and efficient. The algorithm starts with an arbitrary triangulation of the polygon and then deletes a diagonal that leaves only convex components.

Before discussing the algorithm, some terms should be defined. A vertex in a polygon is *reflex* if the internal angle is greater than 180 degrees. A diagonal *d* is *essential* for vertex *v* if removing *d* would create a non-convex subcomponent. A diagonal that is not essential is called *inessential*. Clearly if *d* is essential it must be

incident (i.e. connected) to *v*, and *v* must be reflex (i.e. non-convex). The decision of whether a diagonal deletion will create a non-convex subcomponent can be made locally based on the edges surrounding the diagonal. Any diagonal that does not create a *reflex* is removed.

Hertel and Mehlhorn's algorithm is simply represented as:

triangulate polygon P
while possible do
    remove an inessential diagonal
end

Partitioning the polygon into convex subregions can cause additional lines to be added to the polygon. For our problem, partitioning can produce "artificial" lines that may contain CPs and should be searched. An example is shown in Figure 4–7. The blue lines are the skeleton, and the red line is the "artificial" line added to partition the polygon into two convex subregions.

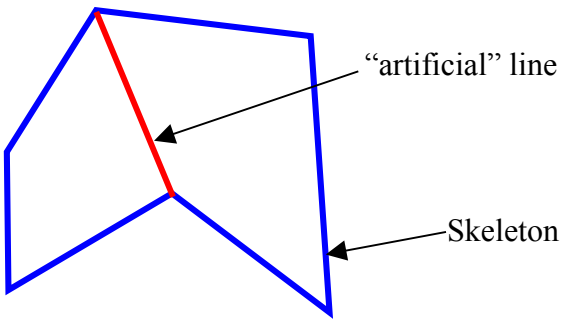


Figure 4–7. Partition of a polygon

We observe that for each convex subregion, the critical values for index-2 CPs are smaller than those of index-1 CPs. Likewise, the critical values of index-1 CPs are smaller than those of index-0 CPs. This provides a guide for searching for the CPs in the order of index-2, index-1, and index-0. This rule applies to all primitives included in this research.

4.2.2 Detailed Algorithm

Normally index-1 CPs are located at the intersection between two primitives, or in the middle of line segments, while index-2 CPs appear inside a convex area. We locate index-2 CPs first. The convex hull is calculated, and then each loop in the convex hull is partitioned to subregions. During this process some "added"/"artificial" lines might be added to form polygons. The flow charts of the algorithms are presented in Figures 4–8 and 4–9. Two examples follow to illustrate how the algorithm works.

4.2.2.1 Searching for Index-2 CPs

Step 1. Find the convex hull. As discussed above, computing the convex hull may result in additional lines added to the skeleton. We call these "added" lines. These lines are stored for future use.

Step 2. Find existing loops in the convex hull. The connectivity of the skeletal elements and the "added" lines determines the number of loops that we need to search.

Step 3. For each loop, check to see if it is convex. If it has one or more concave points, partition the loop. "Artificial" lines may be added as a result of this partitioning. For triangle elements, we only consider

those edges which form boundaries of loops. The other edges of triangles are ignored. Hence, all possible loops are identified.

Step 4. For each possible loop, search for the CPs. Search algorithms used in this step are described below. Loops for which no CPs are found are eliminated by combining them with other loops that do have CPs. The modified loops are stored for later use in topology matching.

At this point, all index-2 CPs are found, and their corresponding search regions are stored. The next step is to locate the index-1 CPs.

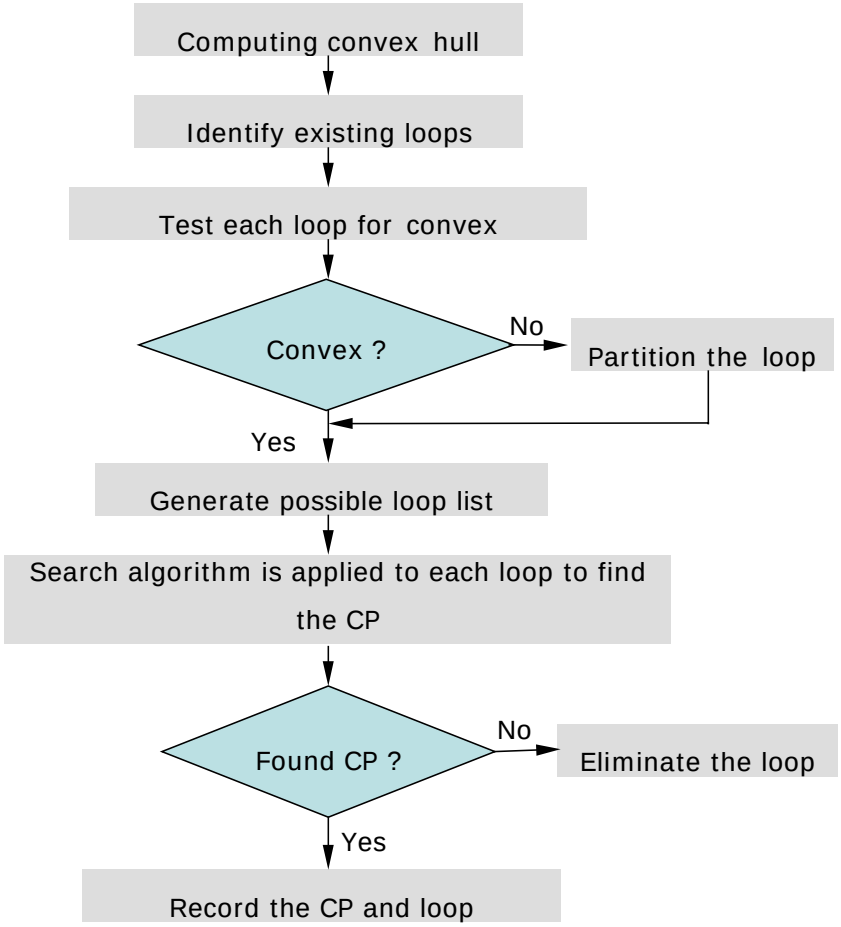


Figure 4–8. Searching for index–2 CPs.

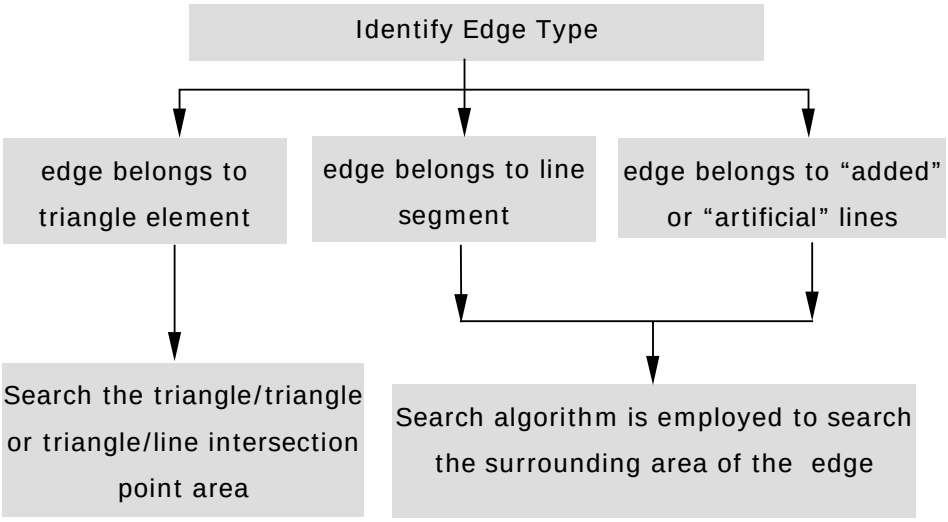


Figure 4–9. Searching for index–1 CPs.

4.2.2.2 Searching for Index-1 CPs

Based on the observations described in section 4.2, we find that index-1 CPs usually appear in the middle of line segments and at the intersection points of triangle elements. These observations hold for the "artificial"/"added" lines as well. The algorithm for locating index-1 CPs is given below.

Step 1. We already know the edge lists from the loops. Each edge in a loop is identified based on which skeletal type it belongs to. For example, we can distinguish edges that belong to triangles from line segments.

Step 2. For those edges belonging to line segments, the search algorithm is employed to find the CP in the surrounding regions of the line segments.

Step 3. For those edges belonging to triangle elements, the index-1 CPs tend to be located near the intersection between two triangle elements or between a triangle and a line segment. In that small region, the search algorithm is applied.

Step 4. For added and artificial lines, we search the surrounding regions in the same manner as line segments.

4.3 SEARCHING ALGORITHM

As described before, locating the critical points of convolution surfaces requires determining the zeros of a non-linear function, i.e., the gradient of the surface equation. This has proven to be intractable in closed form and requires the use of numerical approaches. The search space has been identified for each type of CP. A robust algorithm now must be employed to locate the CPs.

4.3.1 Problem Definition

The problem is to locate the CPs of the convolution surfaces. Recall that the CPs are the points where the first partial derivatives with respect to *x, y,* and *z* vanish simultaneously. This condition can be rewritten as an optimization problem as follows:

$$\begin{cases} \min & ff(x,y,z) = \left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2 + \left(\frac{\partial f}{\partial z}\right)^2, \\ subject\quad to: & \{x,y,z\} \in \{searching\quad space\}. \end{cases} \tag{4–1}$$

where f is the composite convolution surface, and $\partial f / \partial x$, $\partial f / \partial y$, and $\partial f / \partial z$ are the first derivatives of the convolution surfaces respect to *x, y,* and *z.* After casting the search as an optimization problem, an optimization algorithm can be applied.

Gradient-based and non-gradient based search algorithms were explored during the course of this research. In particular, we investigated differential evolution and

sequential quadratic programming. Differential evolution (DE) is a simple population-based stochastic function minimizer using an evolution strategy [SP95, SP96, webDE07]. Sequential quadratic programming algorithm (SQP) is a popular gradient-based optimization algorithm. Both methods can solve the problem defined in equation 4–1, however, SQP is faster compared to DE. Hence, SQP was used to perform the search for CPs.

4.3.2 Sequential Quadratic Programming (SQP)

The SQP algorithm is a powerful method for solving differentiable non-linear programming problems. The basic idea is that, at each step, a local model of the optimization problem is constructed and solved, yielding a step toward the solution of the original problem. The following is an overview of the SQP algorithm.

Consider the general optimization problem of minimizing an objective function *F(x)* under certain constraints, i.e., for $x \in R^n$

$$\begin{cases} \min & F(x), \\ subject \quad to: & g_i(x) \geq 0 \quad i = 1,...,m, \end{cases} \tag{4–2}$$

where *x* is an *n*-dimensional parameter vector.

The SQP algorithm is an iterative method that solves a quadratic programming problem (QP) at each iteration. A quadratic program (QP) is a function that finds the minimum of a quadratic problem subject to linear constraints. The basic idea is to formulate and solve a quadratic programming sub-problem to determine the search direction d_k for each iteration quadratically.

The QP subproblem at iteration *k* is:

$$\begin{cases} \min & \frac{1}{2} d_k{}^T \nabla^2_{xx} L(x_k, \lambda_k) d_k + d_k \nabla L(x_k)^T, \\ subject \quad to: & \nabla g_i(x_k)^T d_k + g_i(x_k) \geq 0, \qquad i = l+1, ..., m, \end{cases} \tag{4–3}$$

where $L(x, \lambda) = F(x) - \sum_{i=1}^{m} \lambda_i g_i(x)$ is the Lagrangian function, and $\lambda = (\lambda_1, ... \lambda_m) \in R^m$ is the multiplier vector. The solution is used to update the solution vector, e.g., $x_{k+1} = x_k + \alpha_k d_k$. The step length parameter α_k is determined by an appropriate line search procedure so that a sufficient decrease in a merit function is obtained. The matrix $\nabla^2_{xx} L(x_k, \lambda_k)$ is a positive definite approximation of the Hessian matrix of the Lagrangian function. Detailed information about the SQP algorithm can be found in the literature (see, e.g., [NW99, Pow77]).

The steps to apply SQP algorithm are:

Step 1. Set $k = 1$.

Step 2. Solve the QP problem described in equation (4–3) to get the solution x_k and Lagrange multiplier λ_k.

Step 3. Compute the step length α_k and set $x_{k+1} = x_k + \alpha_k d_k$.

Step 4. Update the Hessian $\nabla^2_{xx} L(x_{k+1}, \lambda_{k+1})$.

Step 5. Increment k. Stop if $\nabla L(x_k, \lambda_k) \approx 0$; otherwise, go to Step 2.

This method for locating CPs was implemented in the MATLAB® environment. We used the function FMINCON, which uses the SQP algorithm to solve the optimization problem.

® The MathWorks, Inc., Natick, MA.

4.3.3 Computing Time

Most researchers study the topology of implicit surfaces based on point primitives, which is computationally the simplest type. The first derivatives and the second derivatives are not very complicated to compute. When using higher dimensional skeletal elements, the analytic formula for convolution surfaces becomes complicated (see details in Appendix A). Computing the gradient of a convolution surface generated by a higher dimension primitive is even more complicated. Hence, the computational complexity for convolutions of higher dimensional primitives is large compared to those generated with point primitives. The implementation was run on a desktop Pentium® 4 CPU 1.5 GHz PC with 1.0 GB of RAM. Detailed computing times are reported for each case study.

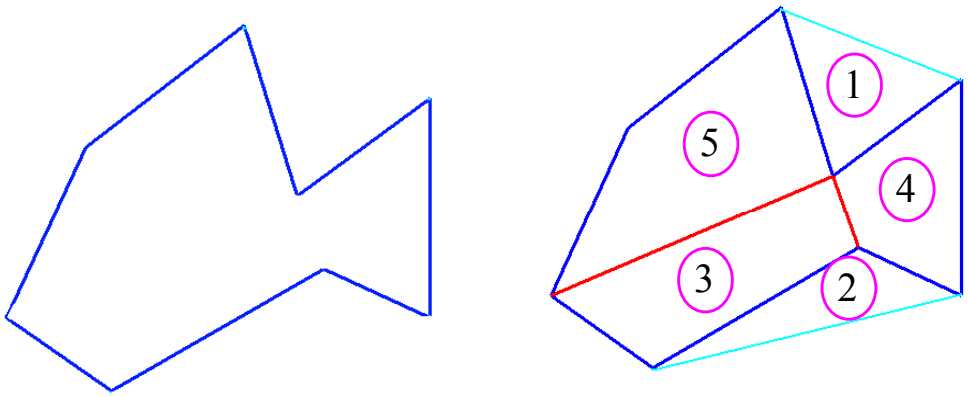
4.4 CASE STUDIES

This section presents two examples to demonstrate the algorithm for finding CPs. We are mainly focused on skeletons with line segments and triangle elements.

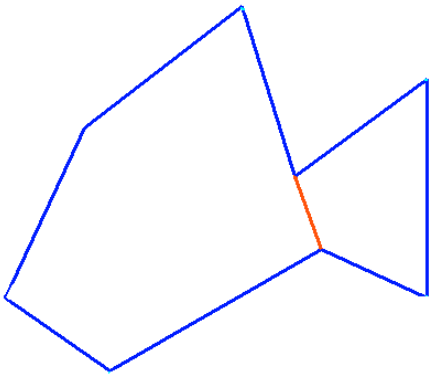
4.4.1 Example One

The skeleton for the first example is shown in Figure 4–4(a). The skeleton is a polygon formed by eight line segments. The width of the Cauchy kernel function s is set to 0.75 for all line segments. Details of the geometry for this skeleton are shown in Table 4–1.

® Intel, Inc., Santa Clara, CA.



(a) skeleton (b) initial partition



(c) partitioned skeleton

Figure 4–10. An example skeleton and its resulting partition.

Table 4–1. Geometry of the skeleton for example one.

Geometry type	Geometry
Line Segment	{0,0,0}→{8,5,0}
Line Segment	{8,5,0}→{12,3,0}
Line Segment	{12,3,0}→{12,12,0}
Line Segment	{12,12,0}→{7,8,0}
Line Segment	{7,8,0}→{5,15,0}
Line Segment	{5,15,0}→{-1,10,0}
Line Segment	{-1,10,0}→{-4,3,0}
Line Segment	{-4,3,0}→{0,0,0}

First the convex hull is calculated. Two lines (green in Figure 4–10(b)) are added. Then the polygon is subdivided into three different convex regions (the two red lines are added). This is the initial partition of the skeleton, shown in Figure 4–10(b). The SQP algorithm is used to calculate the CPs. We find that regions 1, 2, and 3 do not have any CPs. So the added lines in regions 1 and 2 are removed, which are boundary lines (green). CPs are found in regions 4 and 5. The detailed CP information, i.e., critical values, critical types, etc., is stored, as well as the edge lists of the loops. Since region 3 does not have any CPs, this loop should be removed. There are two added lines for the boundary of region 3, so we need to test them to determine which one should be removed. The two lines were added to form the two convex regions 4 and 5, which will be used later to find the index 1 CPs. The distance between each CP and the added lines is calculated. The line with shorter distance to the CPs is removed. Hence, the final partition of the polygon is shown in Figure 4–10(c). The modified loops are stored.

The next step is to determine the index-1 CPs. The surrounding regions of the line segments including the added/artificial lines are searched. The resulting CPs of the convolution surfaces are shown in Figure 4–11. The two red stars are the index-2 CPs, which appear in the middle of the loops. The small circles are the index-1 CPs, which appear near the middle of the line segments.

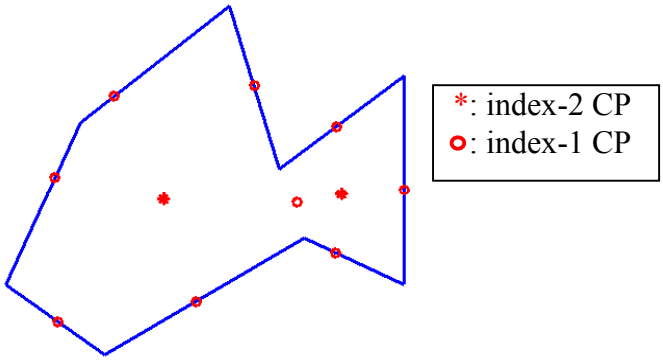


Figure 4–11. CPs of the convolution surfaces from example one.

Table 4–2. CPs of the convolution surfaces from example one.

Critical Value	CP_type	CP_location
0.1368	2	{2.3571,6.6693,0}
0.6234	2	{9.4652,6.9011,0}
0.7784	1	{7.6909,6.5633,0}
2.1122	1	{-2.0076,7.6406,0}
2.1182	1	{3.6421,2.2787,0}
2.1223	1	{0.3600,11.1239,0}
2.1594	1	{-1.8962,1.4416,0}
2.2004	1	{11.9821,7.0920 0}
2.2268	1	{5.9788,11.5681,0}
2.2859	1	{9.2461,4.3778,0}
2.3451	1	{9.2785,9.8113,0}

The critical values, types, and locations of the CPs are shown in Table 4–2. Two index-2 CPs and 9 index-1 CPs were found. The time for computing the CPs for this case is approximately 180 seconds.

4.4.2 Example Two

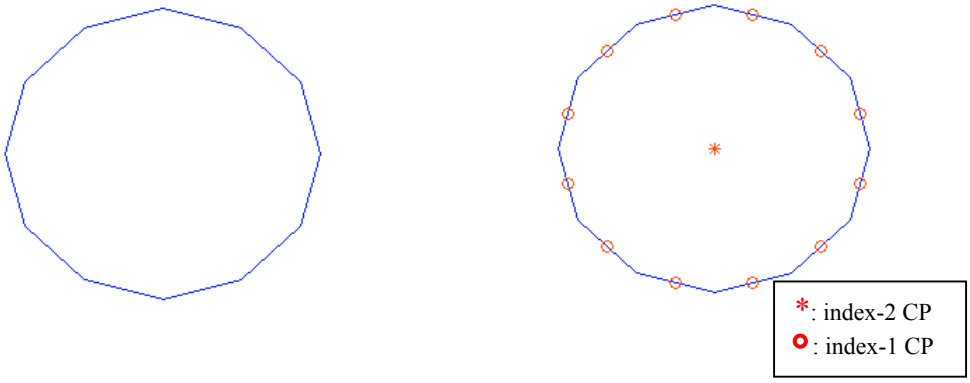
The next example is a circle approximated by 12 line segments. All end points of the line segments are located on the circle. The geometric information is listed in Table 4–3. The width *s* of the Cauchy kernel is set to 0.8 for this example. The skeleton and the CPs of the generated convolution surfaces are shown in Figure 4–12. The detailed information about the CPs is given in Table 4–4.

Table 4–3. Geometric information for the skeleton of example two.

Geometry type	Geometry
Line Segment	{10,0,0}→{8.7,5,0}
Line Segment	{8.7,5,0}→{5, 8.7,0}
Line Segment	{5,8.7,0}→{0,10,0}
Line Segment	{0,10,0}→{-5,8.7,0}
Line Segment	{-5,8.7,0}→{-8.7,5,0}
Line Segment	{-8.7,5,0}→{-10,0,0}
Line Segment	{-10,0,0}→{-8.7,-5,0}
Line Segment	{-8.7,-5,0}→{-5, -8.7,0}
Line Segment	{-5,-8.7,0}→{0, -10,0}
Line Segment	{0,-10,0}→{5, -8.7,0}
Line Segment	{5,-8.7,0}→{8.7,-5,0}
Line Segment	{8.7,-5,0}→{10,0,0}

Table 4–4. CPs of the convolution surfaces for the ring.

Critical Value	CP_type	CP_location
0.016	2	{0,0,0}
1.9704	1	{-9.3568,2.4634,0}
1.9704	1	{9.3569,-2.4633,0}
1.9704	1	{-2.4633,9.3569,0}
1.9704	1	{9.3569,2.4633,0}
1.9704	1	{2.4633,9.3569,0}
1.9704	1	{-9.3569,-2.4632,0}
1.9704	1	{-2.4631,-9.3569,0}
1.9704	1	{2.4631,-9.3569,0}
1.9704	1	{6.8482,6.8482,0}
1.9704	1	{-6.8482,-6.8482,0}
1.9704	1	{-6.8482,6.8482,0}
1.9704	1	{6.8482,-6.8482,0}



(a) skeleton (b) location of CPs

Figure 4–12. Skeleton and CPs of example two.

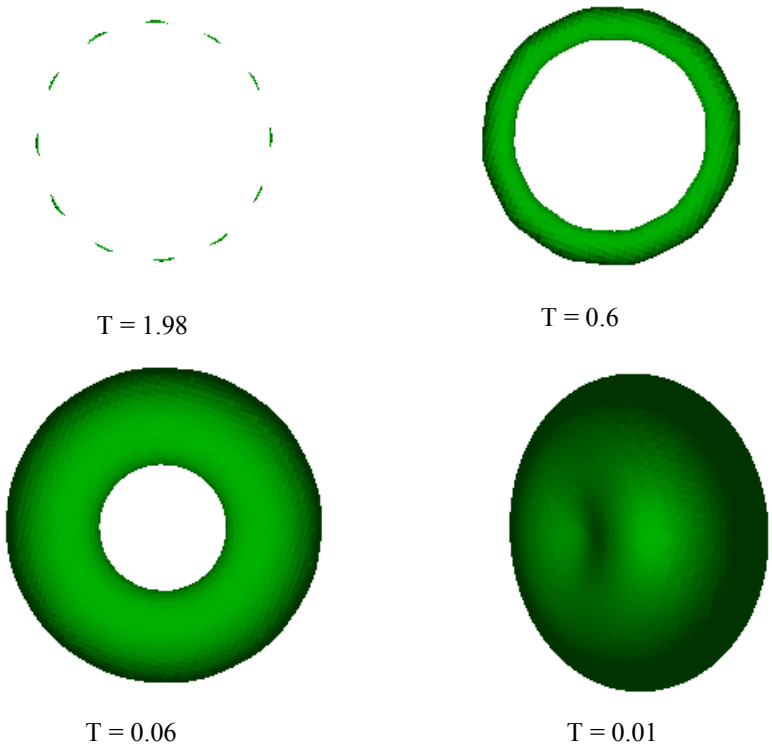


Figure 4–13. Generated convolution surfaces with different threshold values *T*.

The convolution surfaces with different threshold values *T* are shown in Figure 4–13. We notice that the critical value of the index-2 CP is very small for this case. When checking the eigenvalues, they are also relatively small (-0.0013, -0.0013, and 0.0007). However, none of the eigenvalues is zero. When using other kernel functions, the common influence areas[10] are relative small, so the CP in the middle could not be found, which is the case that is illustrated in Wu and Malheiros [WM99]. They search for the CPs in the common influence areas only, while the center point is located outside these areas. However, the Cauchy kernel function has the property that it vanishes when *r*

[10] As described in Chapter 2, the common influence areas are those areas where the values of the function are not uniformly zero.

approaches infinity. One problem that appears is that the CP has a tendency to become degenerate when the skeletal primitives are placed far from each other. In this case, Morse theory breaks down.

The time to compute the CPs for this case is 270 seconds. This is longer compared to the first case since this example contains more skeletal primitives.

4.5 SUMMARY

Skeleton-based modeling is a simple method for generating approximate shape during conceptual design of a product. We propose that a skeleton represents not only the shape of the product, but also the topology of the final shape. The location of the CPs of the generated surfaces should be related to the skeleton. This chapter discusses an algorithm for finding the CPs based on the properties of the skeleton. The index-1 CPs are found mostly between pairs of points and pairs of triangle elements. For line segments, the CPs are found near the center location of the line segments. For index-2 CPs, we look for closed polygons formed by line segments in the skeleton. Each concave polygon of the skeleton is partitioned into several convex areas. Each area has at most one index-2 CP. The SQP algorithm is then employed to find the CP in each region. The examples show that the proposed algorithm appears to be effective at finding the CPs.

After finding all CPs, the topology of the generated surfaces can be analyzed by using Morse theory. The next chapter presents how the topology of the surface changes when the threshold value passes a critical value. A strategy for choosing a threshold value that matches the topology of the generated surfaces with that of the skeleton is also presented in Chapter 5.

Chapter 5 Topology Analysis of Convolution Surfaces

In Chapter 4, the algorithm for locating the critical points of convolution surfaces was discussed. In this chapter the topology of convolution surfaces is determined based on Morse theory. Invariants, namely the Betti numbers, are used to determine the topological features of a surface. These invariants are calculated for skeletons to determine the proper topology of the convolution surfaces generated. Our approach to changing the topology of convolution surfaces to match that of the skeleton is the main topic of this chapter.

5.1 TOPOLOGICAL INVARIANTS

Recall from Chapter 3 that two kinds of topological invariants are used to measure the topology of the surfaces: the Euler characteristic and Betti numbers. The Euler characteristic χ is related to the surface genus, *g*, which is the number of holes in a surface. Betti numbers consist of three individual numbers: β_0, β_1, and β_2. The first, β_0, is the number of gaps that separate components, that is, the number of connected components; β_1 is the number of tunnels that pass through the shape; and β_2 is the number of voids that are components of the complement space inaccessible from the outside.

Recall that the topology of a surface refers to the connectedness of the shape, which includes the number of disconnected components, the number of tunnels/holes, and the number of voids in each component. When comparing the topology of convolution surfaces with that of its root skeleton, we are not only interested in the number of holes, but also in the number of voids and the number of components. Betti

numbers provide rich topological information about surfaces; hence in this research, Betti numbers are used to characterize the topology of skeletons as well as convolution surfaces.

5.2 TOPOLOGY OF THE SKELETON

A *skeleton* is a connected set of geometric primitives (in our case, points, line segments or triangles). Given a set of skeletal primitives and their connectivity, the Euler characteristic is easily calculated through equation 3–1. Then the genus can be easily computed by equation 3–2. The genus only represents the number of holes in the skeleton.

- Computing β_0

Recall that in chapter 4, in discussing the designer's intent, we assume that all skeletal elements are connected, i.e., there is only one object. If there is more than one object in the skeleton, we will process each individually. Therefore, the number of connected components of the convolution surfaces, β_0, is always 1. Our main job is to calculate the other two Betti numbers.

- Computing β_2

Betti number β_2 is the number of closed regions in space, and can be computed by determining whether all components are closed. For the skeleton itself, only triangle primitives can form closed regions in space. Therefore, we only focus on triangle primitives. The easiest way to determine whether a component is closed is to perform a depth first search from each triangle to each of its neighboring triangles. If one of the triangle's edges does not have a neighbor, then the skeleton touches the boundary, and the surface is not closed. For each void area formed, the corresponding triangle elements are stored to record the location of the closed region.

- Computing β_1

Betti number β_1 can be calculated based on equation 3–3 once the Euler characteristic χ and the values of β_0 and β_2 are calculated. This gives the number of loops in the skeleton. But this is global topological information; it does not indicate the locations of the loops. Two objects with the same Betti numbers are topologically the same. However, when we look at the local topology of the objects, they may not match. In conceptual design, each loop or feature of an object exists for certain desired functionality. These features are included to meet engineering requirements. The generated convolution surfaces should also have loops or features in the same general locations. Hence, we need to know the exact location of each loop so that the topology of the convolution surfaces can be compared with the topology of the root skeleton. The connectivity between each skeletal element can be retrieved by the halfedge data structure discussed in Chapter 4.

5.3 TOPOLOGY MATCHING

The algorithm for determining the threshold value T such that the topology of the composite convolution surfaces matches that of the underlying skeleton is based on analyzing the convolution surfaces using Morse theory, as described in the previous chapter. Morse theory indicates when and where the topology changes. We notice that when the threshold value T for the convolution surfaces is very small, the contributions from each of the skeletal elements to the composite surfaces will blend together. Since we assume the skeleton is one object (i.e., all skeletal elements are connected), we start analyzing topology changes from very small values of T, when the object is a single blended lump without holes or voids. As mentioned in Chapter 4, we observe that the critical values for index-2 critical points (CPs) are generally smaller than those of index-1 CPs. Likewise, the critical values of index-1 CPs are smaller than those of index-0 CPs.

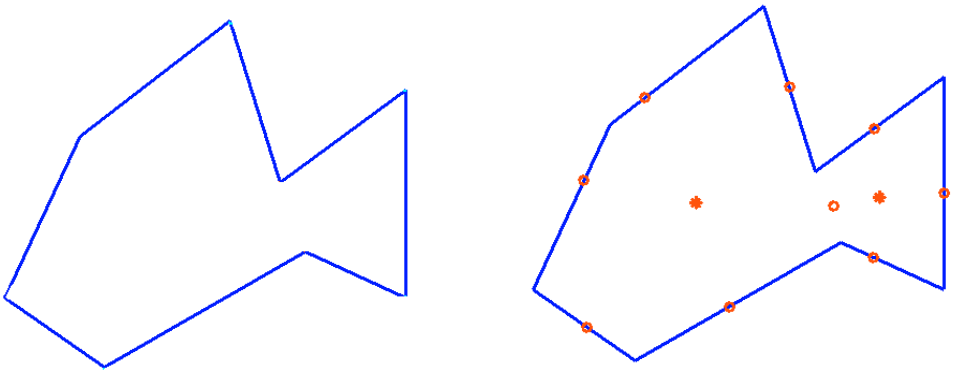
The CPs are sorted based on their critical values in ascending order. When the threshold value *T* gradually increases, it passes through all critical values in order. Every time the value of *T* passes a critical value, the topology of the composite convolution surfaces changes. When the value of *T* falls in the interval between two critical values, the topology of composite convolution surfaces remains the same for all values in that interval.

```
Number_of_objects = 1;
Number_of_loops = 0;
Number_ of _voids = 0;

for i=1 to number_of_critical_points –1
      if sorted_critical_point[i].type == 3
            Number_of_voids += 1;
            Modify void list;
      if  sorted_critical_point[i].type == 2
            Number_of_loops += 1;
            Modify loop list;
            If Number_of_voids > 0 && associated loop belongs to voids_list
                  Number_of_voids -= 1;
            Endif
      Else if sorted_critical_point[i].type == 1
            if number_of_loops >0
                  if associated connecting location belongs to a loop
                        number_of_loops -= 1;
                        modify loop list;
            else
                  number_of_objects +=1;
            endif
      endif

//check whether the topology is the same as the skeleton, output the T data range
      if  Number_of_loops == skeleton_loops &&
        Number_of_objects ==skeleton_objects
        Output: sorted_critical_point[i] and sorted_critical_point[i+1].
      End if
End for loop
```

Figure 5–1. Algorithm to calculate the topology of composite convolution surfaces.

For example, an index-2 CP indicates that a loop forms around the CP. An index-1 CP indicates that a connection somewhere is broken. Depending on the location of the CP, two different topological changes can occur: (a) the object is divided into two objects, or (b) a loop is broken around the CP location. If the skeletal elements associated with the index-1 CP do not belong to a loop, the first topology change occurs. The algorithm then stops comparing the CPs, since only one object is considered at a time (i.e, index-0 CPs are ignored). If the skeletal elements belong to a loop, the number of loops is reduced by one, which is the second case. If the skeletal element belongs to two loops i.e., it is a common skeletal element of two loops, then the two loops are combined into one loop, and the loop list is modified. The number of objects remains the same. Details of the algorithm are shown in Figure 5–1.

The example in Chapter 4 is used to demonstrate the topology changes. The skeleton and the CPs for the generated convolution surfaces are repeated in Figure 5–2. The critical values, types, and locations of the CPs of the convolution surfaces are listed in Table 5–1, sorted by the critical values.



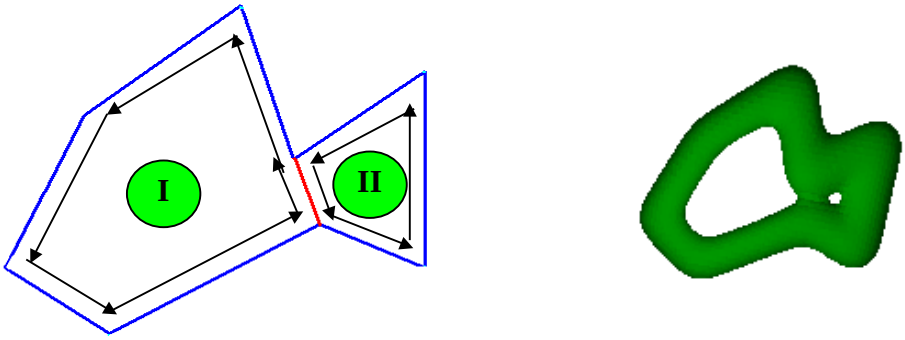
(a) the skeleton (b) CPs of the generated convolution surfaces

Figure 5–2. An example of topology matching.

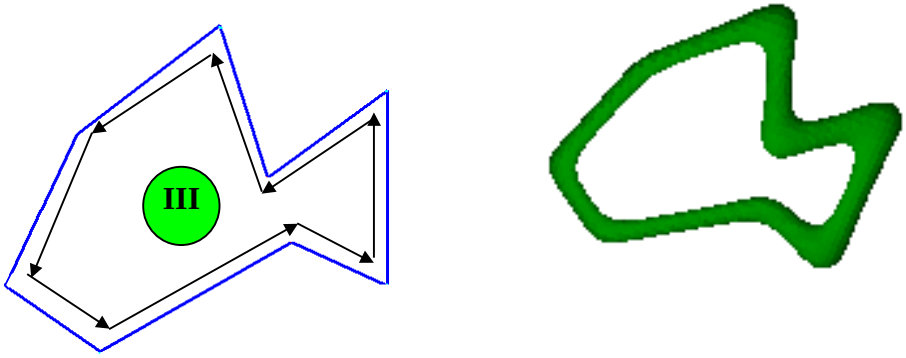
Table 5–1. CPs of the convolution surfaces

Critical Value	CP type	CP_location
0.1368	2	{2.3571,6.6693,0}
0.6234	2	{9.4652,6.9011,0}
0.7784	1	{7.6909,6.5633,0}
2.1122	1	{-2.0076,7.6406,0}
2.1182	1	{3.6421,2.2787,0}
2.1223	1	{0.3600,11.1239,0}
2.1594	1	{-1.8962,1.4416,0}
2.2004	1	{11.9821,7.0920,0}
2.2268	1	{5.9788,11.5681,0}
2.2859	1	{9.2461,4.3778,0}
2.3451	1	{9.2785,9.8113,0}

In this example, when T passes 0.1368, which is the critical value of an index-2 CP, a loop appears. This increases β_1 to 1, and loop I forms, as shown in Figure 5–3(a). When T reaches the second critical value of 0.6234, which is another index-2 CP, a second loop appears (loop II), and β_1 increases to 2. When T reaches 0.7784, the critical value of an index-1 CP, a different topology change occurs. The skeletal element associated with the CP, which is the red line shown in Figure 5–3(a), belongs to the common edge of the two loops. The two loops are merged to form one big loop (III), shown in Figure 5–4(b), and β_1 changes back to 1. The skeletal element related to the next CP is the red line shown in Figure 5–3(c). When T passes this value (2.1128), which is the critical value of an index-1 CP, loop III is broken and no loops are left. Betti number β_1 changes to 0, while β_0 remains 1. As T continues to increase, it reaches the next critical value 2.1223, another index-1 CP. The object breaks into two components, and β_0 increases to 2, as shown in figure 5–4(d). The topology analysis stops at this point, since more than one object appears.

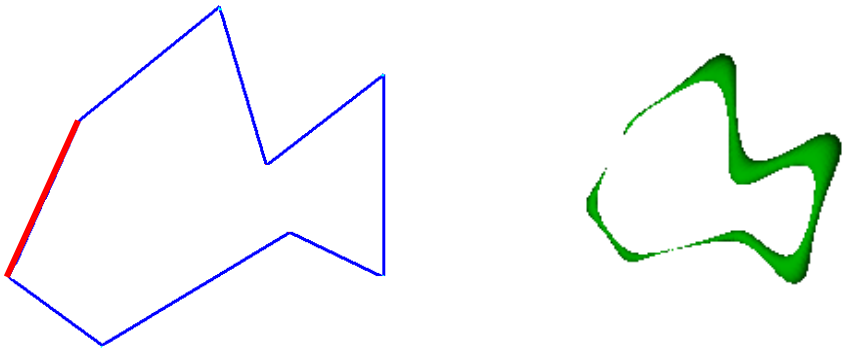


(a) Two loops and the common edge (red) between them, and its corresponding convolution Surfaces

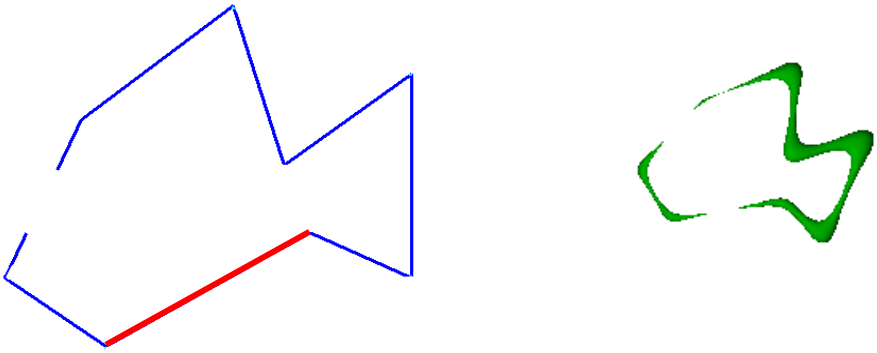


(b) one loop and its corresponding convolution surfaces

Figure 5–3. Topology analysis of example in Figure 5–2.



(c) the location (red line) where the loop is broken, and the convolution surfaces



(d) the location (red line) where the object is broken to two objects, and the convolution surfaces

Figure 5–3 (continued). Topology analysis of example in Figure 5–2.

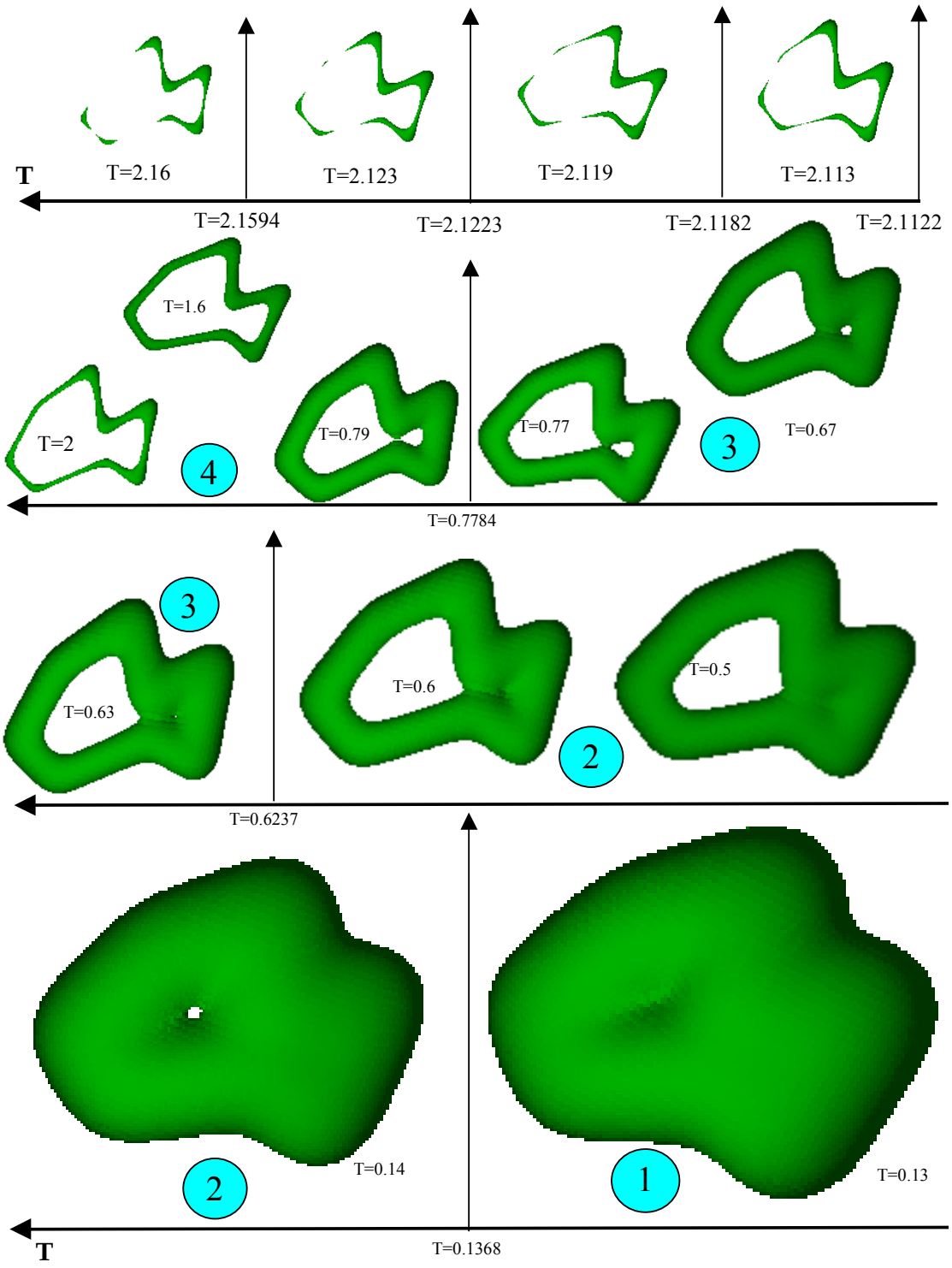


Figure 5–4. Relationship between topology changes and threshold value *T*.

Table 5–2. Topology changes of convolution surfaces.

T	Topology	Topology Changes
T<0.1368	β_0=1; β_1=0; β_2=0	One lump object
0.1368<T<0.6234	β_0=1; β_1=1; β_2=0	A loop appears
0.6234<T<0.7784	β_0=1; β_1=2; β_2=0	A loop appears
0.7784<T<2.1122	β_0=1; β_1=1; β_2=0	A loop is broken
2.1122<T<2.1182	β_0=1; β_1=0; β_2=0	A loop is broken
2.1182<T<2.1223	β_0=2; β_1=0; β_2=0	An object is divided to two
2.1223 <T<2.1594	β_0=3; β_1=0; β_2=0	An object is divided to two

Table 5–2 shows the results of the topology analysis for the first seven critical points, when *T* reaches 2.1594. Three objects form when the value of *T* is between 2.1223 and 2.1594. The generated convolution surfaces with different threshold values *T* are shown in Figure 5–4. The figure clearly shows the relationship between topology changes and the threshold value *T*. As noted above, when the threshold value falls into the interval between two critical values, the topology of the surfaces remains the unchanged.

5.4 MATCHING THE TOPOLOGY OF CONVOLUTION SURFACES AND SKELETON

In order to match the topology of the composite convolution surfaces with the generating skeleton, we compare the Betti numbers of the convolution surfaces and the skeleton. We know the exact number of objects, tunnels, and voids of the skeleton. Based on this information, we can choose the proper value for *T* that will force the generated convolution surfaces to have the same topology as that of the skeleton. For example, the skeleton shown in Figure 5–2 is a single object and has a single loop. Table 5–2 shows that there are two cases that match this topology, characterized by β_0=1, β_1=1, and β_2=0. These cases are represented by the intervals [0.1368, 0.6234] and [0.7784, 2.1122]. When *T* falls in either of these intervals, the topologies of the composite convolution surfaces and the skeleton are equivalent. These cases are illustrated in intervals (2) and (4) in

Figure 5–4. However, the final generated shapes are slightly different. In interval (2) the small open area on the right side of the skeleton is filled. In interval (4), the composite convolution surfaces more closely follow the skeleton, which we assume is closer to the designer's intent. We need to distinguish these two cases.

Recall that when calculating the topology of the skeleton, the relationships between the skeletal elements and topological features, e.g., loops and voids, are explicitly represented. When searching for the CPs of the convolution surfaces, the search boundary is also represented. The search boundary consists of skeletal elements as well as lines added during convex decomposition of the skeleton. For the surfaces to match the skeleton topologically, the particular regions where loops or voids form are compared. In the example above, the loop in interval (2) (Figure 5–4) is formed by loop I (Figure 5–3(a)), while the loop in interval (4) (Figure 5–4) is formed by loop III in Figure 5–3(a). The skeleton has only one loop, which is exactly the same as loop III. Hence interval (4) has the expected shapes. The proper threshold value of T will be in the interval [0.7784, 2.1122].

Sometimes the Betti numbers do not match, or the locations of loops or voids of the surfaces and skeleton do not match. Then the convolution surfaces should be modified.

5.4.1 Distinguishing Different Topologies

From the topology of the skeleton we have the exact number of objects, tunnels, and voids in the object. Once we have determined the topology of the composite convolution surfaces, we can choose the proper threshold value. Three distinct cases may occur:

(1) There is one T value range that produces a match.

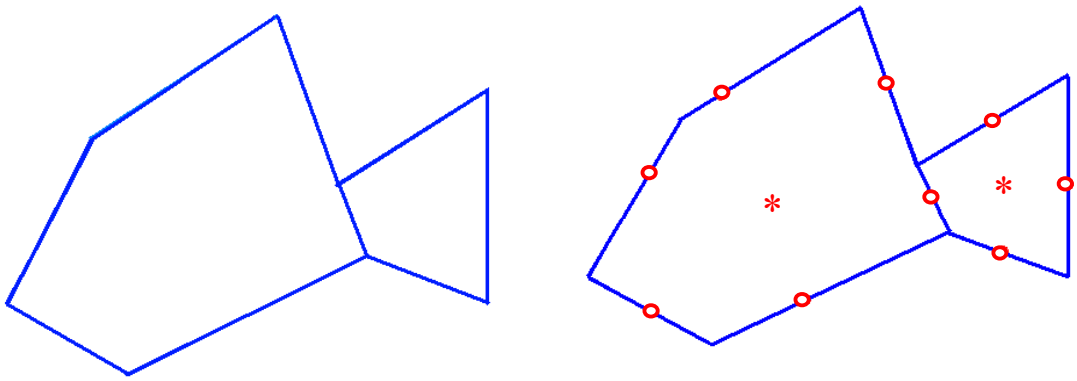
(2) There is more than one T value data range that produces a match.

(3) There is NO proper *T* value to match the skeleton's topology.

We discuss each of these situations below.

Case 1: One T value data range matches the topology of the skeleton

This is the ideal situation. We do not need to change anything and we can provide the proper *T* value to user so that he/she can generate the matching surfaces. The example below is similar to the case shown in Figure 5–2, with an extra line segment added to divide the skeleton into two loops. The skeleton and the locations of the CPs of the convolution surfaces are shown in Figure 5–5.



(a) the skeleton

(b) the CPs for the generated surfaces.
(red circle: index-1 CPs, red *: index-2 CPs.)

Figure 5–5. Case 1: The skeleton and its CP locations.

the Betti numbers are: 1 2 0
the number of objects is: 1
the number of loops is: 2
the number of voids is: 0
the range of threshold value T is: 0.7927 2.1133

**********the related loops are:**********
0 0 0 8 5 0
8 5 0 7 8 0
7 8 0 5 15 0
5 15 0 -1 10 0
-1 10 0 -4 3 0
-4 3 0 0 0 0
****
8 5 0 12 3 0
12 3 0 12 12 0
12 12 0 7 8 0
7 8 0 8 5 0
****

Figure 5–6. Results from analyzing case 1.

The result is shown in Figure 5–6. There is only one threshold value *T* for which the generated convolution surfaces have the same Betti numbers as the skeleton. However, we cannot say that this is the correct value until we determine whether the two loops formed by the line segments match the loop list of the skeleton. If they match, then we can say that is the proper value of *T*. Otherwise, no proper *T* value can be provided by the system (case 3, described below). Recall that, when we calculate the topology of the convolution surfaces, we know the edge list for each loop associated with the surfaces. We also know the edge lists of each loop for the skeleton. Therefore, we can compare these loops. For this example, the loops of the convolution surfaces match those of the skeleton. Hence, once the value of *T* falls into the recommend interval [0.7927, 2.1133],

the generated convolution surfaces are guaranteed to have the same topology as the skeleton.

Case 2: More than one T value data range matches the topology of the skeleton

The example in section 5.3 is used to demonstrate this case. The skeleton and its CPs are shown in Figure 5–2. The results are shown in Figure 5–7.

the number of objects is: 1
the number of loops is: 1
the number of voids is: 0
****
the range of threshold value T is: 0.1368 0.6234

**********the related loops are:**********
0 0 0 8 5 0
8 5 0 7 8 0
7 8 0 5 15 0
5 15 0 -1 10 0
-1 10 0 -4 3 0
-4 3 0 0 0 0

****
the range of threshold value T is: 0.7784 2.1122

**********the related loops are:**********
7 8 0 5 15 0
5 15 0 -1 10 0
-1 10 0 -4 3 0
-4 3 0 0 0 0
0 0 0 8 5 0
8 5 0 12 3 0
12 3 0 12 12 0
12 12 0 7 8 0

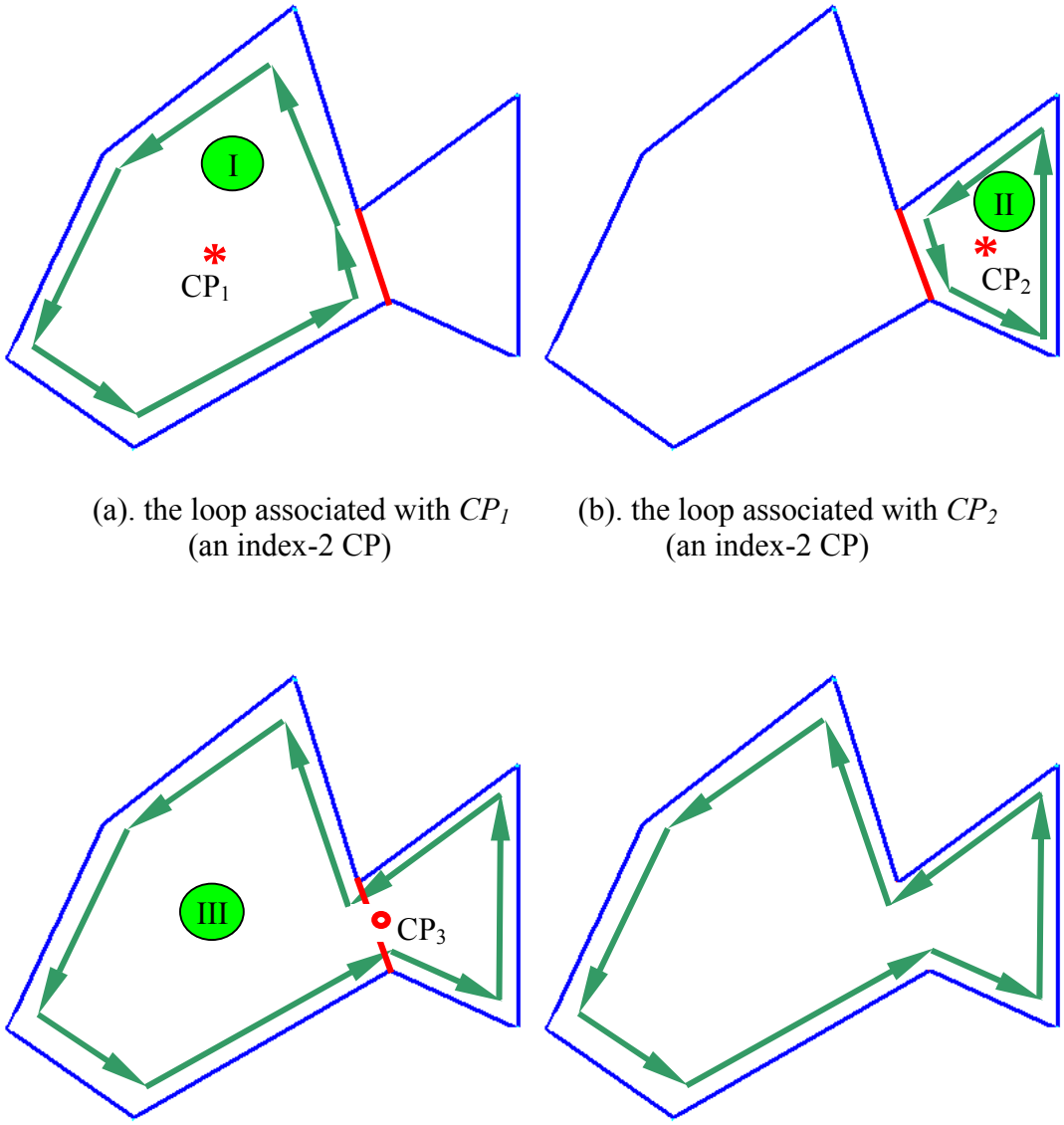
****

Figure 5–7. Results from analyzing case 2.

The skeleton has a single loop. Two ranges of threshold values are provided by the system to match the topology of the surface with that of the skeleton. When the value of T falls between 0.1368 and 0.6234, a small loop forms. However, the area to the right side of the skeleton is filled. When T falls between 0.7784 and 2.1122, one large loop forms. We can see that the two generated shapes are different. The system should determine which has the intended geometry.

Similar to case 1, the loop lists of the convolution surfaces are compared to the loops of the skeleton. For the first range, the loop is associated with the critical point CP_1, which is an index-2 CP. The loop is shown in region I in Figure 5–8(a). Note that this loop includes one "artificial" edge that is a result of the convex decomposition of the original skeleton, and is the loop used to bound the search for critical points. We can see that the search loop does not match the original skeletal geometry (Figure 5–8(d)).

Now we consider the second option, the larger search loop (region III in Figure 5–8(c)). This loop is related to CP_3, which is an index-1 CP. Before the threshold value increases to the critical value at CP_3, it passes the critical value at CP_2, where another loop around CP_2 in region II is formed, as shown in Figure 5–8(b). Now we can analyze the larger loop. Associated with CP_3 is an "artificial" edge. When the threshold value T passes the critical value of CP_3, the connection is broken at the critical point location. Therefore a one large loop is formed (region III in Figure 5–8(c)). This loop is identical to that of skeleton. We can say that the convolution surfaces now have the desired geometry. In this case, the recommend threshold value T provided to the designer is the interval [0.7784, 2.1122]. Once T falls into that interval, the generated convolution surfaces have the same topology as well as the same shape with the root skeleton.



(a). the loop associated with CP_1 (an index-2 CP)

(b). the loop associated with CP_2 (an index-2 CP)

(c). a big loop formed associated with CP_3 (an index-1 CP)

(d). The loop in the skeleton.

Figure 5–8. Topology match analysis for case 2.

Case 3:No proper T value can be found.

In this case, the CPs are calculated, but no matching topology can be found. An example skeleton is shown in Figure 5–9(a), consisting of four line segments. The geometry and property information are showned in Table 5–3, where *s* controls the width of the Cauchy kernel function. The critical point information in shown in Figure 5–9(b).

Table 5–3. The geometry information and the value of *s*.

Line Segment Location	**Value of *s***
{0,0,0}→{5,0,0}	0.5
{5,0,0}→{5,1,0}	0.5
{5,1,0}→{0,1,0}	0.5
{0,1,0}→{0,0,0}	0.5

(□ : index-0 CP)

(a) Skeleton (b) CPs of generated convolution surfaces

Figure 5–9. Skeleton and CPs for case 3.

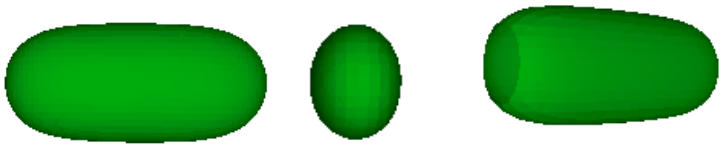


Figure 5–10. Generated convolution surfaces.

In this case, only one CP is found, an index-0 CP. Since no index-2 CP is generated for the convolution surfaces, no loop is generated based on the current parameter settings. However, the skeleton has one loop. Regardless of the threshold value T, the topology of the convolution surfaces will not match the topology of the root skeleton. Either the convolution surfaces or the skeleton must be modified for a match to occur.

5.4.2 Modifying the Skeleton and its Properties

In cases where an appropriate threshold value cannot be determined, other parameters must be considered to realize a topological match between the skeleton and the composite convolution surfaces. Several parameters determine the shape and topology of convolution surfaces, including:

- Geometry of the skeleton.
- Kernel functions. Different kernel functions can generate different shapes as well as different topologies for convolution surfaces.
- The width (s) of the Cauchy kernel function, which controls the shape of convolution surfaces.
- Threshold value (T). T is critical to the topology of the convolution surfaces since it determines the level set.

Modifying the skeleton itself can change the topology of the convolution surfaces. However, we assume that only the designer should change the skeleton, since the designer initially created the skeleton according to his/her intent. The system will not modify the skeletal geometry. The system should only modify properties of the convolution surfaces themselves. The kernel function is one candidate for modification. However, not all kernel functions provide analytical solutions for all skeletal elements

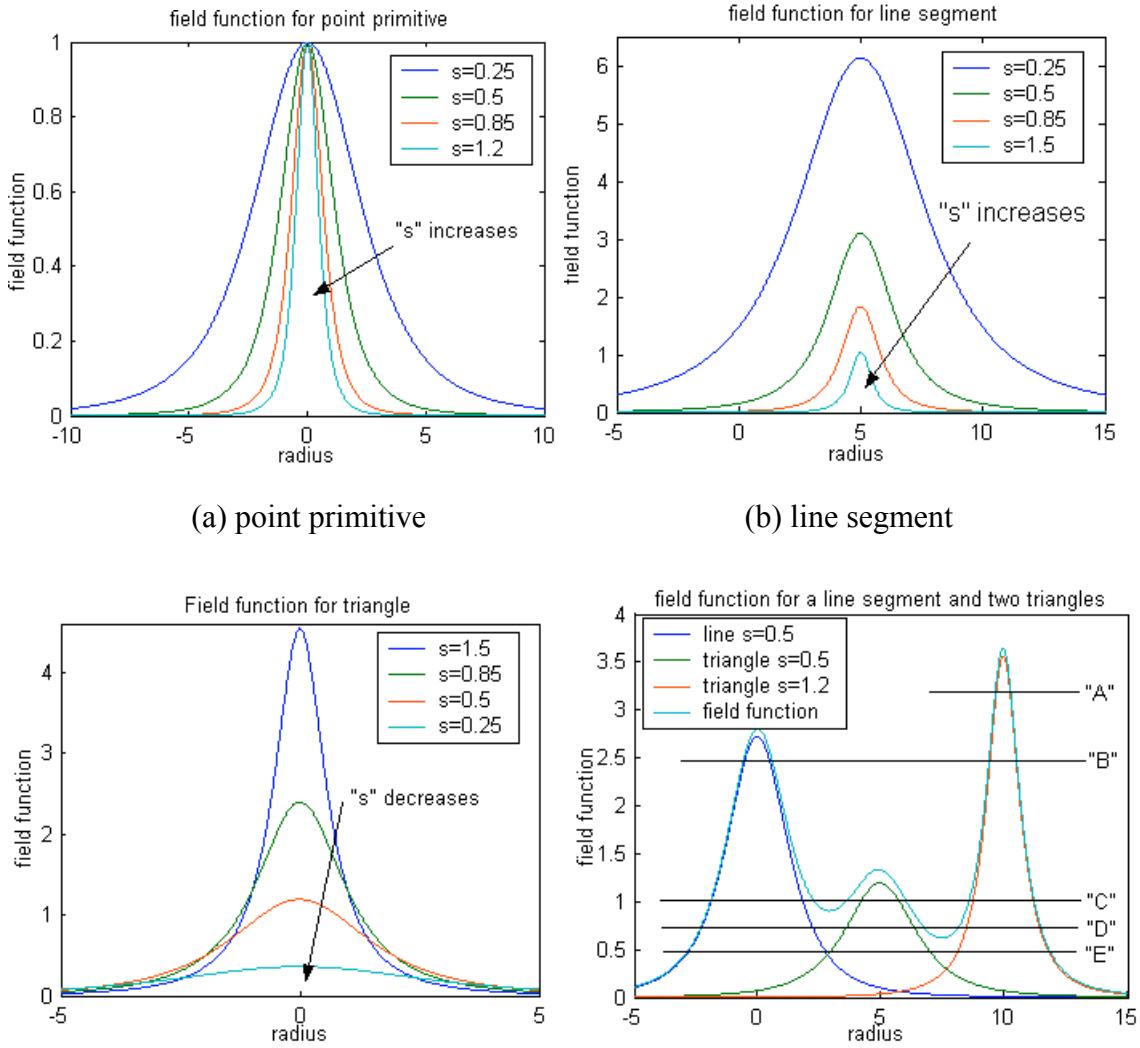
used in our system. This is one reason we chose the Cauchy function in the first place. Hence, two parameters are left as candidates for modification. These are the width of the kernel function *s* and the threshold value *T*. Threshold values are recommend by the system based on the critical values and the type of CPs. The particular value of *T* will not change the topology of the convolution surfaces within the data range recommended by the system and cannot be used to remedy case 3 situations. Hence only the value of *s* can be changed to modify the topology in these situations.

5.4.3 Parameter *s* – Width of the Cauchy Kernel Function

The value of *s* affects the radius of the skeletal elements. Generally, the smaller the value of *s*, the wider the generated convolution surfaces. This is true for all geometric primitives included in this system. However, there are differences among the primitives. Point primitives have very desirable properties. As the value of *s* increases, the width of the field function decreases, but the maximum value of the field function remains constant. Figure 5–11(a) shows that that the threshold value is always less than 1. This is not true for line segments and triangle elements. These differences increase the difficulty in devising a strategy for adjusting the kernel function width. For the line segment primitive, increasing *s* decreases the field function width, but also results in a decrease in the maximum value of the field function (Figure 5–11(b)). For smaller values of *s*, the radius of the generated surfaces increases, and larger values of *T* can be selected. For the triangle element, increasing *s* decreases the field function width, similar to the other primitives. However, for this primitive, increasing *s* results in an increase in the maximum value of the field function (Figure 5–11(c)).

Figure 5–11(d) shows a case with one line segment and two triangle elements. When the value of *T* is larger, say at level A, only the contribution from the right side triangle element appears. When *T* decreases to level B, the contribution from the line

appears. When T decreases to level C, the contribution from the second triangle appears. Since the value of s for the second triangle is small (s = 0.5), the maximum field function value is much lower than that for the other two skeletal primitives. Basically the generated convolution surfaces have a fairer shape when the value of s is large for triangle elements. So when adjusting the value of s, different rules should be applied to different skeletal primitives.



(a) point primitive (b) line segment

(c) triangle primitive (d) one line segment and two triangles

Figure 5–11. Relationship between field function and width s.

5.4.4 Adjusting the Value of *s*

We are interested in adjusting the value of *s* for the case where there is a loop in the skeleton but the loop is filled by the generated convolution surfaces. This situation occurs when the two skeletal elements are placed very close, and the geometry of one skeletal primitive dominates the other, or when *s* is not assigned a proper value. The convolution surfaces generated by the two skeletal elements appear as a single object for all values of *T*. No index-2 CP occurs. Figure 5–12 shows the skeleton of two line segments that are placed very close to each other, with $s = 0.2$ for line 1 and $s = 0.65$ for line 2. Figure 5–13(a) shows the field functions for the two close line segments shown in Figure 5–12, *"f1"* with $s = 0.2$, and *"f2"* with $s = 0.65$. We know from the topology of the skeleton that the two elements are separated. If we knew the radius of the generated convolution surfaces for each skeletal element, this would help us choose the proper value for *s*. However, the radius of the generated convolution surfaces is determined by the geometry and the values of *T* and *s*. Both *T* and *s* are uncertain at this stage of the process. It is difficult to determine the value of *s*, and better strategy may be to let the user choose the value. The user may choose different values of *s,* or he/she may modify the skeleton so that proper surfaces can be generated.

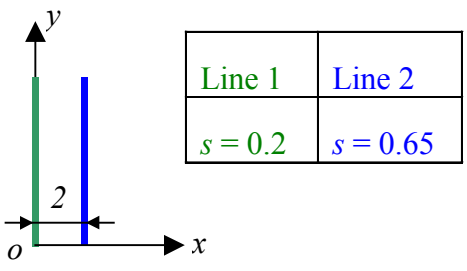
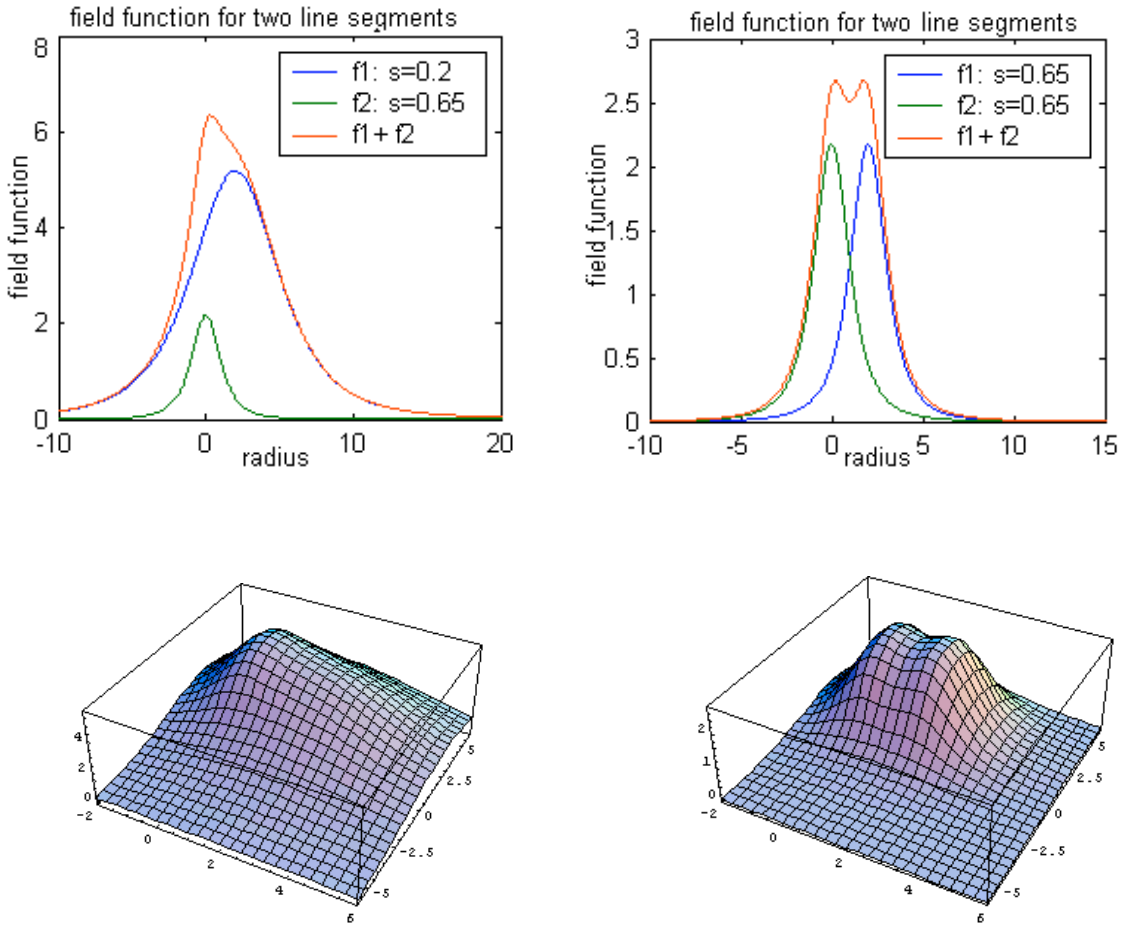


Figure 5–12. Skeleton of two close line segments and their widths.



(a) Connected (b) Separated for certain value of T

Figure 5–13. Field function for two line segments close to each other.

If the two line segments are connected, an easy way to separate them is to make them thinner while keeping the same geometry. No quantified values of s can be provided as discussed above, but some heuristics can be used. The values of s should be increased

so that thinner surfaces can be generated. If the values of *s* are different for the two line segments, the smaller value of *s* can be increased to the larger one. Then a value of *T* can be selected to disconnect the two line segments. In Figure 5–10, when the value of *s* for *"f1"* increases to the same value as that of *"f2"*, $s = 0.65$, the two line segments are disconnected for certain values of *T* (Figure 5–13(b)). For triangle elements, a similar approach is taken, i.e., increasing the value of *s* provides the opportunity to select larger threshold values of *T* so that the generated surfaces are thinner.

For the example of case 3 in section 5.4, we can modify the value of *s* to make the generated surface thinner and form a loop. From Figure 5–11(b) we can see that, for line segments, when the value of *s* increases, the width of the kernel function decreases. For this example, when the value of *s* increases from 0.5 to 1.5, a loop appears when *T* falls in the interval [1,0661, 1.2311]. Convolution surfaces with a threshold value $T = 1.15$ are shown in Figure 5–14.

Figure 5–14. Modified convolution surfaces for skeleton in Figure 5–8.

5.5 SUMMARY

In this research we characterize the topology of convolution surfaces as well as skeletons by the Betti numbers, since they provide more complete topological information. Morse theory is an effective tool for analyzing the topology of smooth

surfaces such as convolution surfaces. When the threshold value of the convolution surfaces reaches the critical values, topology changes can be determined by the CP types. The locations of connection breaks loop disappearances can also be determined by the relationships of the critical point locations to the underlying skeletal geometry. This can provide guidance for modifying the convolution surfaces when necessary. Examples of our proposed skeletal modeling system are discussed in next chapter.

Chapter 6 Case Studies

This chapter presents three examples to demonstrate the topology analysis and matching algorithms presented previously. Section 6.1 presents a bottle opener example previously demonstrated by Thompson [Thom00]. Section 6.2 demonstrates an approach to incorporating tetrahedral elements in the skeletal modeling scheme. In section 6.3, a realistic example of designing a steering wheel is presented.

6.1 BOTTLE OPENER

The bottle opener example below is borrowed from [Thom00] to demonstrate skeletal modeling and topology analysis with convolution surfaces. The skeleton of the bottle opener is formed by 18 points in [Thom00]. In our work, the skeleton is modeled with 9 line segments (blue) and two triangles (yellow), as shown in Figure 6–1. Based on the algorithm presented in Chapter 4, we calculate the CPs first. Secondly, the topologies of the convolution surfaces and the skeleton are determined. Finally the topology match strategy is applied. The recommend threshold value of T is calculated. For this example, we will analyze two different cases: 1) a proper value of T is recommend by the system; and 2) no proper value of T is recommend by the system and the property values of the generated convolution surfaces must be adjusted.

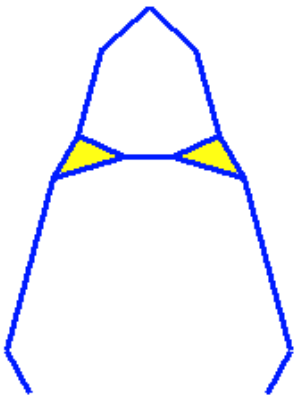


Figure 6–1. Skeleton of the bottle opener.

6.1.1 Case 1

The geometry for the bottle opener is shown in Table 6–1. The width of the kernel function is set to 0.95 for both line segments and triangle elements.

Table 6–1. Geometry information for the bottle opener.

Element Type	Coordinates
Line	{0,-4,0}→{ -1, -2, 0}
Line	{-1, -2, 0}→{1, 6,0}
Line	{2, 8, 0}→{3, 12,0}
Line	{3, 12, 0}→{5, 14, 0}
Line	{5, 14, 0}→{7, 12, 0}
Line	{7, 12, 0}→{8, 8, 0}
Line	{9, 6, 0}→{11, -2, 0}
Line	{11, -2, 0}→{10,-4,0}
Line	{4, 7, 0}→{6, 7, 0}
Triangle	{2, 8, 0}→{1, 6, 0}→{4, 7, 0}
Triangle	{8, 8, 0}→{6, 7, 0}→{9, 6, 0}

The partitioned skeleton and CPs are shown in Figure 6–2. The steps for calculating the critical points are as follows:

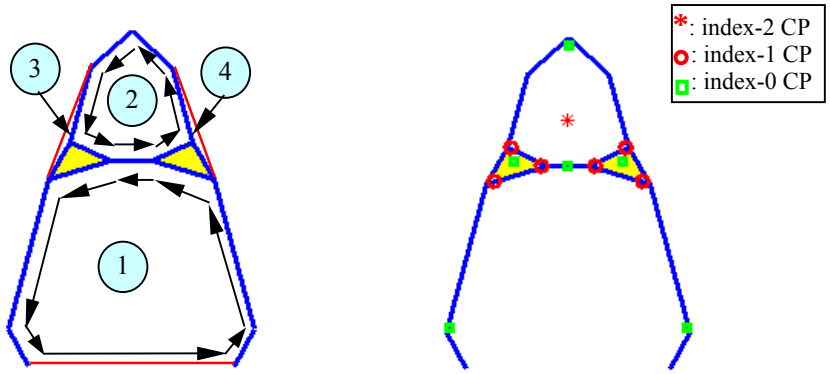
Step 1. Calculate the convex hull based on the skeletal geometry information. Three lines (shown in red) are added to form the convex hull.

Step 2. Partition the convex hull into convex subregions. There are four convex subregions, shown in Figure 6–2(a).

Step 3. In each convex subregion, apply the SQP algorithm to determine the index-2 CPs. After calculation, only one CP is found in subregion 2. No CPs are found in the other three subregions.

Step 4. Calculate the index-1 CPs.

Step 5. Calculate the index-0 CPs.



(a) Partitioned skeleton (b) CPs of the convolution surfaces

Figure 6–2. Partitioned bottle opener skeleton and CPs of generated convolution surfaces.

The last step is to calculate the index-0 CPs. The areas that possibly contain index-0 CPs as described in Chapter 4 are searched. The purpose for this step is threefold: 1) to show that the system has the ability to locate index-0 CPs, even though we do not consider index-0 CPs for most cases; 2) to show that the location of the different types of

CPs changes with the skeletal elements; and 3) to demonstrate the entire topology change process associated with the changes of the threshold value *T*.

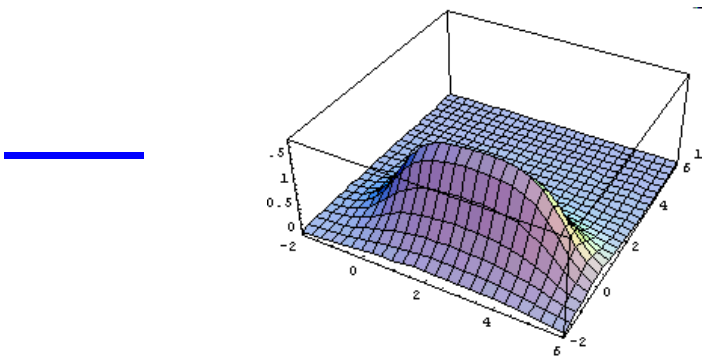
For this example, 13 CPs are found: one index-2 CP, six index-1 CPs, and six index-0 CPs. These is illustrated in Figure 6–2(b). The critical point information is listed in Table 6–2.

Table 6–2. Critical values of the bottle opener.

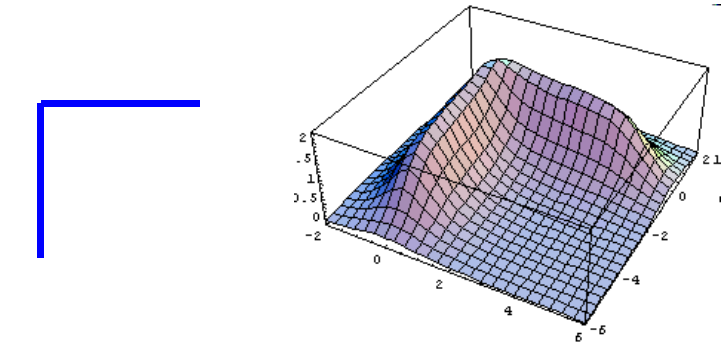
Critical_Value	Type	Location
0.2422	2	{5.0000, 9.5771, 0}
1.1086	1	{3.6700, 7.0570, 0}
1.1086	1	{6.3300, 7.0570, 0}
1.1131	1	{8.6363, 6.1896, 0}
1.1131	1	{1.3637, 6.1896, 0}
1.478	1	{2.1841, 7.9629, 0}
1.478	1	{7.8159, 7.9629, 0}
1.4839	0	{7.6903, 7.2273, 0}
1.4839	0	{2.3097, 7.2273, 0}
1.5107	0	{5.0000, 7.0069, 0}
1.684	0	{ -0.8505, -1.8128, 0}
1.684	0	{10.8505, -1.8128, 0}
1.9671	0	{5.0000, 13.5931, 0}

In analyzing the data, we notice that each CP's location is different from the corresponding CP of the example in Chapter 4. The CPs move around the surface when the skeleton geometry changes. Figure 6–3 illustrates this phenomena with the field function displayed for three different configurations of skeletal elements. The figure shows the field function at the z=0 plane. In Figure 6–3(a) the maximum point is in the middle of the line segment. When two line segments share a common vertex, the location of the maximum point changes to the intersection point of the line segments, as shown in Figure 6–3(b). Figure 6–3(c) shows a configuration with four lines forming a loop. Note that saddle points appear in the middle of the line segments, and another type saddle point

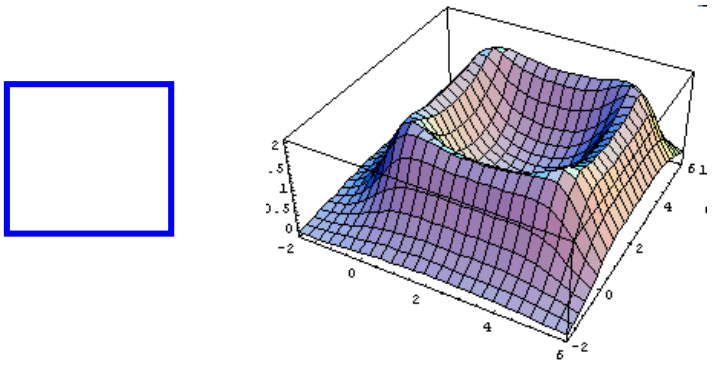
appears in the middle of the loop formed by the four line segments. Hence, with different skeletal settings, the locations of CPs float around the space.



(a) one line segment



(b) two connected line segments



(c) four connected line segments

Figure 6–3. Field functions for line segment skeletons with different configurations at z=0 plane.

- *Topology of the skeleton*

The skeleton is a single object, hence $\beta_0 = 1$. There is a loop in the top of the skeleton, which is formed by the top line segments and one edge from each of the triangle elements. The loop is illustrated in Figure 6–4 with black arrows. Hence, $\beta_1 = 1$. The loop is recorded for later use to identify the loop location. There are no voids in the skeleton, so $\beta_2 = 0$.

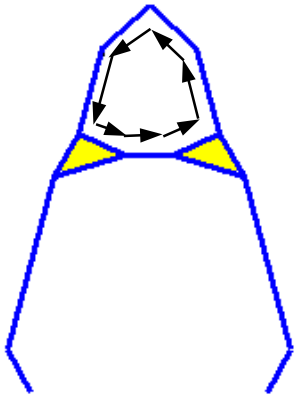


Figure 6–4. Loop in the bottle opener skeleton.

- *Topology of the convolution surfaces*

As the threshold value changes, the topology of the convolution surfaces changes. When the threshold value is very small, the whole object is a single lumped spheroid, as shown in Figure 6–5. The Betti numbers are $\beta_0 = 1$, $\beta_1 = 0$, and $\beta_2 = 0$. When the value of T increases to the critical value 0.2422, a loop is formed in region 2, and the topology of the generated convolution surfaces changes. The Betti numbers become $\beta_0 = 1$, $\beta_1 = 1$, and $\beta_2 = 0$. As the value of T continues increasing, the connection between the line segments and the triangle elements is broken, and the topology of the generated surfaces

changes to $\beta_0 = 3$, $\beta_1 = 1$, and $\beta_2 = 0$. At this point the skeleton produces three disconnected objects. When the value of T reaches 1.1131, the connections between the triangle elements and the middle line segment are broken, resulting in a break in the loop. The Betti numbers change to $\beta_0 = 4$, $\beta_1 = 0$, and $\beta_2 = 0$.

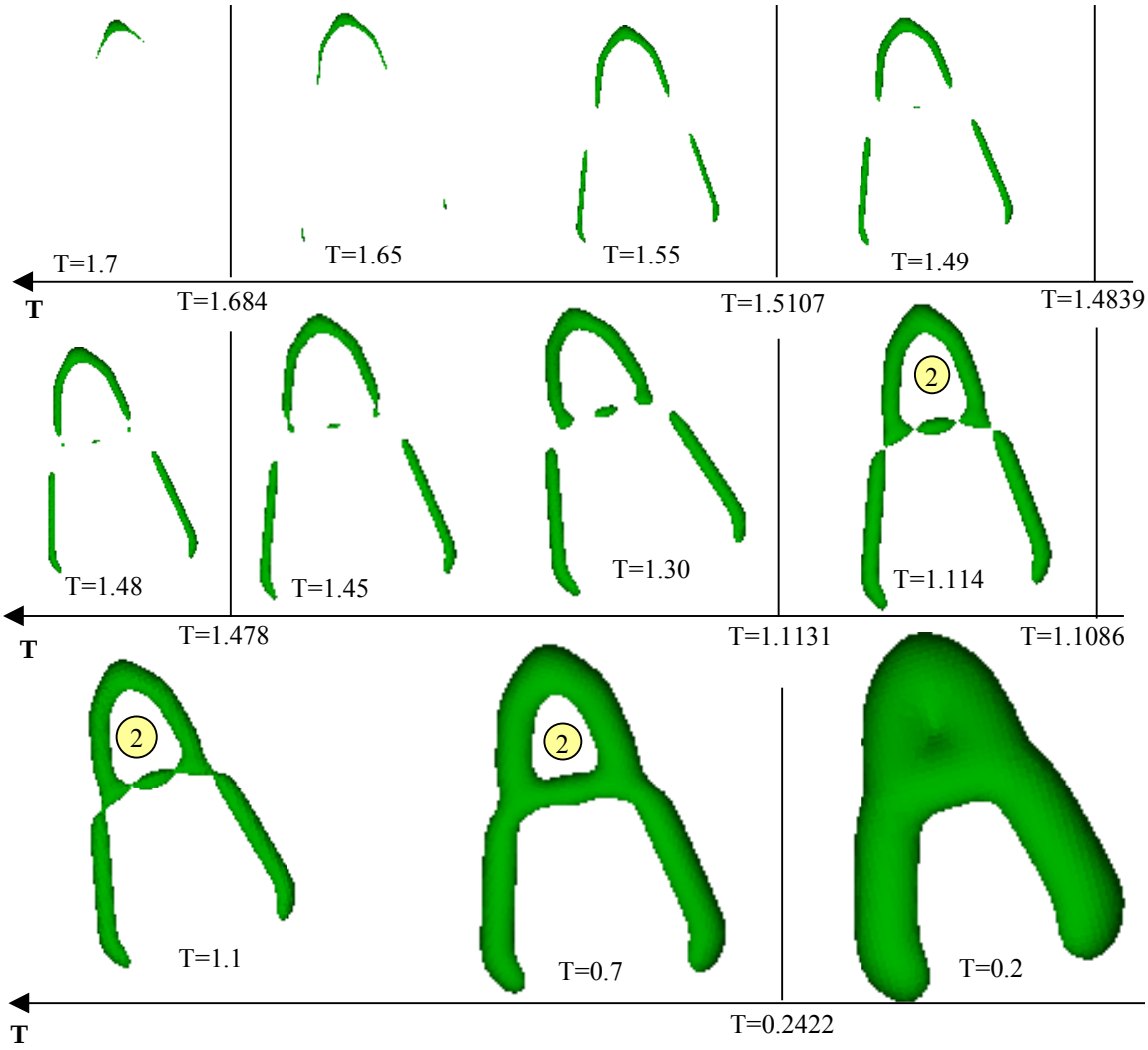


Figure 6–5. Topology changes for the bottle opener for different threshold values.

The only range that produces a topology match between the convolution surfaces and the skeleton (i.e., $\beta_0 = 1$, $\beta_1 = 1$, and $\beta_2 = 0$) is [0.2422, 1.1086]. The next step is determining whether the loop is in the same location for the surfaces and the skeleton. For the surface, the loop is formed in region 2 shown in Figure 6–2(a), which is the same loop as the skeleton. Thus, the recommended threshold value range for this case is [0.2422, 1.1086]. This is demonstrated in Figure 6–5.

6.1.2 Case 2

This example demonstrates a case for which no proper value of T exists, and as a result, the parameter values of the convolution surfaces must be adjusted. We continue with the bottle opener as an example. The skeleton has the same geometry as case 1; the only difference is the width s of the kernel function is set to 0.45 for all skeletal elements. Since the value of s is small, the generated convolution surfaces are "fatter" compared to those generated with a larger s value. The critical value for the index-2 CP is 2.14, hence the loop can only be generated with a threshold value $T > 2.14$.

The CPs found with this setting are different from those found in case 1. The index-0 CP associated with the triangle element cannot be found. Instead, an index-1 CP is found. The triangle elements are dominated by the line segments. In order to make the triangle elements appear, we need to change the value of s. Recall from Chapter 5 that, for a triangle element, the smaller the value of s, the larger the radius of the generated convolution surfaces. However, the maximum value of the field function is also small. With a larger threshold value T, no surfaces will be generated. When the value of s is larger, the generated convolution surfaces are thinner, but the maximum value of the field function for the triangle element is larger. Settings for s in this range generate surfaces with larger values of T. Hence, we increase the value of s so that the triangle elements

can contribute to the surfaces. In order to keep the loop, the threshold value T should be larger than 2.1, which is the critical value of the index-2 CP. We choose $T = 2.2$, then calculate the proper value of s for the triangle element. Considering a single triangle in the skeleton, the relationship between s and T is shown in Figure 6–6. If T is set to 2.2, the corresponding value of s should be larger than 1.35. However, considering the influence of the other line segments, we can choose a smaller value of $s = 1.0$. Figure 6–7 shows these different configurations of the bottle opener.

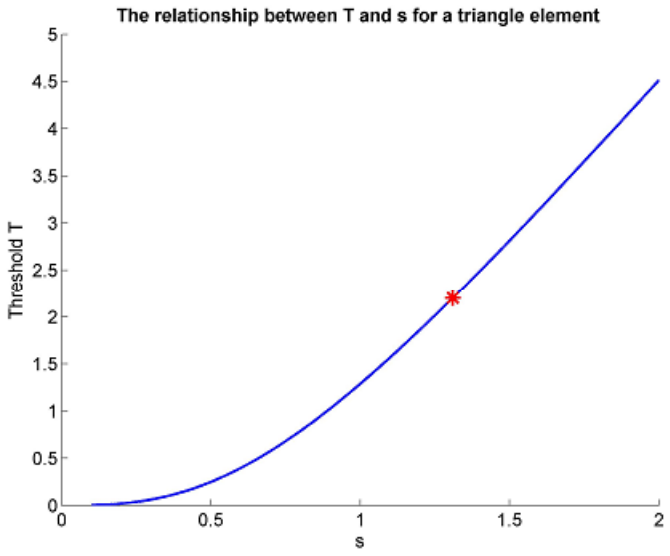


Figure 6–6. The relationship between T and s.

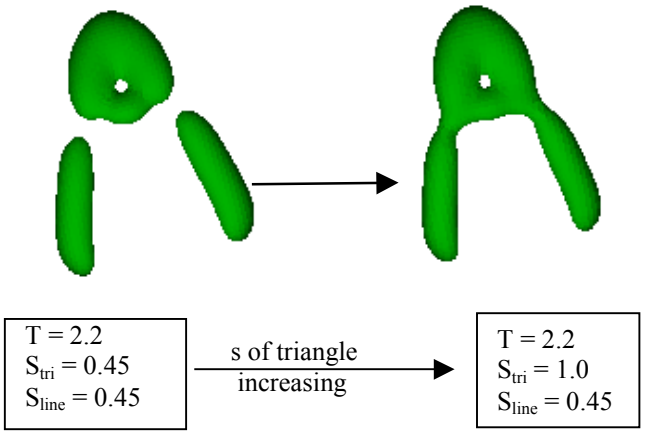


Figure 6–7. Two different configurations of the bottle opener.

The value of *s* affects the topology of the convolution sufaces for certain threshold values *T*. As the value of *s* gradually increases, the maximum value of the field function for the triangle element also increases. This behavior allows the triangle to appear before the other two lines connect. So the idea is that all objects should appear first, and then we will consider whether they are connected or not. From the above example, we can see that the width of the Cauchy kernel *s* cannot be set too small, since the generated convolution surfaces will have a large radius, and certain skeletal elements will be dominated by others.

6.2 TETRAHEDRAL ELEMENTS

We have analytical formulas for convolution surfaces with the Cauchy kernel for point, line segment, and triangle skeletal elements. However, an analytic solution for the tetrahedral element is not known for the Cauchy kernel. If other polynomial kernel functions are used to generate convolution surfaces (e.g., the soft object kernel), an analytical solution can be found for tetrahedral elements, but the generated convolution surfaces are spheroidal shapes and do not resemble the underlying tetrahedron.

One approach that simulates a true tetrahedral element is to use four triangle elements to generate the surfaces. However, it is possible that the generated convolution surfaces result in an inner void. Figure 6–8 shows two wireframe images of convolution surfaces generated with triangular elements in a tetrahedral arrangement. Figure 6–8(a) shows a convolution surface for a tetrahedron that has smooth outer surfaces and no inner void, while Figure 6–8(b) shows a void area generated inside the surfaces.

(a) without inner void (b) inner void generated

Figure 6–8. The convolution surfaces generated based on four triangles formed into a tetrahedral element.

The tetrahedral element is a solid object and should not have an inner void. We can detect the void by analyzing the topology of the convolution surfaces formed by the four triangle elements. An index-3 CP indicates the presence of a void area. By setting the threshold T to a value less than the critical value at the index-3 CP, the will be no void inside the surfaces and the topology of the generated convolution surfaces will match that of the tetrahedron. The range of T values is suggested by the modeler. Once the T is set to a value in the recommend interval, the four triangle elements will effectively be a tetrahedron without an inner void.

Each skeletal element has a threshold value used to generate the convolution surfaces. Recall that Morse theory indicates that an index-0 CP signifies the creation or removal of a component. In order to generate the convolution surfaces, the threshold value should be less than the critical value at the index-0 CP. For example, for a point element (sphere), there is an index-0 CP in the center of the sphere. Once the threshold value is greater than the critical value at the CP, no surfaces are generated. For the simulated tetrahedral element, we can instead set the critical value at index-3 CP as the minimum threshold value to indicate whether the surfaces will be generated.

If the value of T is greater than the recommend range, the generated surfaces are thinner and an inner void area will be generated. We can use other elements, for example, a small sphere, to fill the void. Our main purpose is to find a small surface with no inner void to fill in any void area inside the four triangles. In this way, we can ensure that the generated convolution surfaces have the same topology as the tetrahedron.

The following example demonstrates the proposed simulated tetrahedron. The geometry of the tetrahedron is shown below in Figure 6–9.

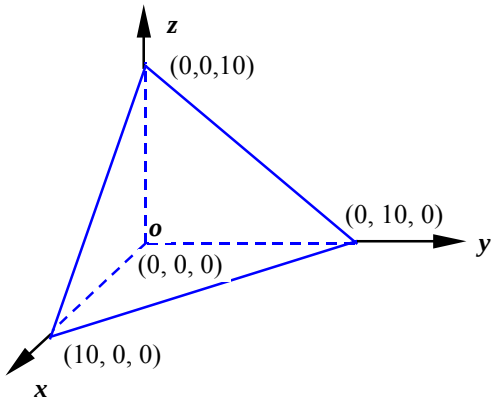


Figure 6–9. Geometry of the tetrahedron element.

The initial value for s is set to 0.85. We are interested in whether there are void regions inside the generated convolution surfaces, so we need to search for index-3 CPs. The search region is the whole space enclosed by the four triangles. For the case shown in Figure 6–8, one index-3 CP is found. The eigenvalues are {-0.2131, -0.2131, -0.0458}, and the CP location is at {2.2113, 2.2113, 2.2113}. The critical value is V_{cp} = 1.71. Hence when the threshold value T is greater than V_{cp}, there will be an inner void. As long as the threshold value T is less than V_{cp}, the surface will not contain an inner void (Figure 6–10).

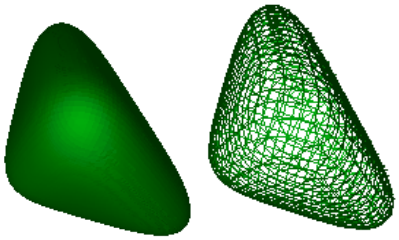
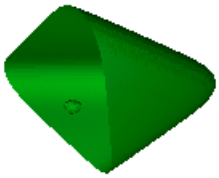
T < 1.71	T >1.71
	
T = 1.0. (left: solid, right: wireframe)	T = 1.75, surface sectioned at x = 2.

Figure 6–10. Convolution surfaces of simulated tetrahedron with different values of T.

In order to accommodate larger values of T, we can fill any void areas with simple skeletal elements so that the topology of the generated surfaces is the same as that of the tetrahedron. The simplest way is to fill the interior with a sphere at the CP location. We need to choose the radius of the sphere so that it will fill the whole space without affecting the outside surfaces. This increases computing time, since we need to calculate the index-2 CPs to ensure that the surfaces generated by the triangle elements do not have loops. The critical values at the index-2 CPs for the example are 2.9945 and 2.9897

(shown in Figure 6–10). After T reaches 2.9897, a hole appears in the plane formed by (10,0,0), (0,10,0), (0,0,10). When T increases to 2.9945, three more holes appear in the other 3 planes, and four loops emerge. Hence, the maximum desirable value of T is less than 2.9897. When T lies in the extended interval [1.71, 2.9897], one or more spheres need to be inserted. For this case, we can add one sphere so that the four triangles still generate a closed composite surface.

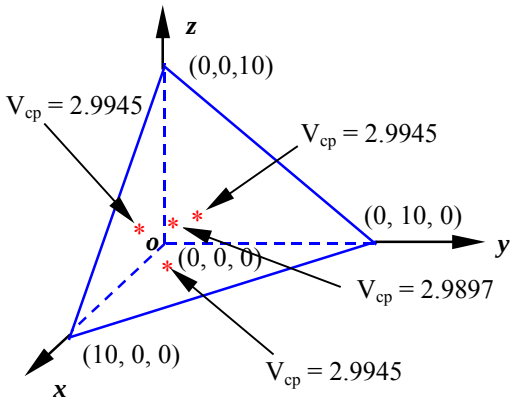
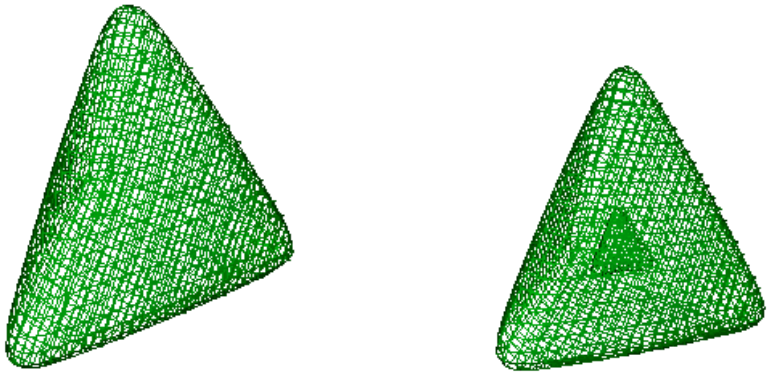


Figure 6–11. Index-2 CPs for example tetrahedron.

Recall from Figure 5–9(a) that, for a single point primitive, the maximum value of the field function remains 1 for any value of s. The field function's radius is larger when the value of s is smaller. The smaller value of s is chosen so that it can cover more space. If T is much bigger than V_{cp}, several spheres must be added. Otherwise the void will not be completely filled. For this example, after adding one sphere (s = 0.25), the value of V_{cp} increases to 2.71, while the surfaces still contain no loops. If T continues to increase, it may be possible that the void is not completely covered by the sphere, or the convolution surfaces may contain loops. For this example, we will focus on the case of adding a

single sphere. This is demonstrated in Figure 6–12. In Figure 6–12(a), when T = 2.7 (which is less than the V_{cp}), no void is generated. As shown in Figure 6–12(b) when T = 2.8 (larger than V_{cp}), a void emerges. In this way, once the threshold value T < V_{cp} (V_{cp} = 2.71), the generated surfaces do not contain an inner void, as expected.



(a) T = 2.7, one sphere (s = 0.25) (b) T = 2.8, one sphere (s = 0.25)

Figure 6–12. Convolution surfaces of simulated tetrahedron after adding one sphere.

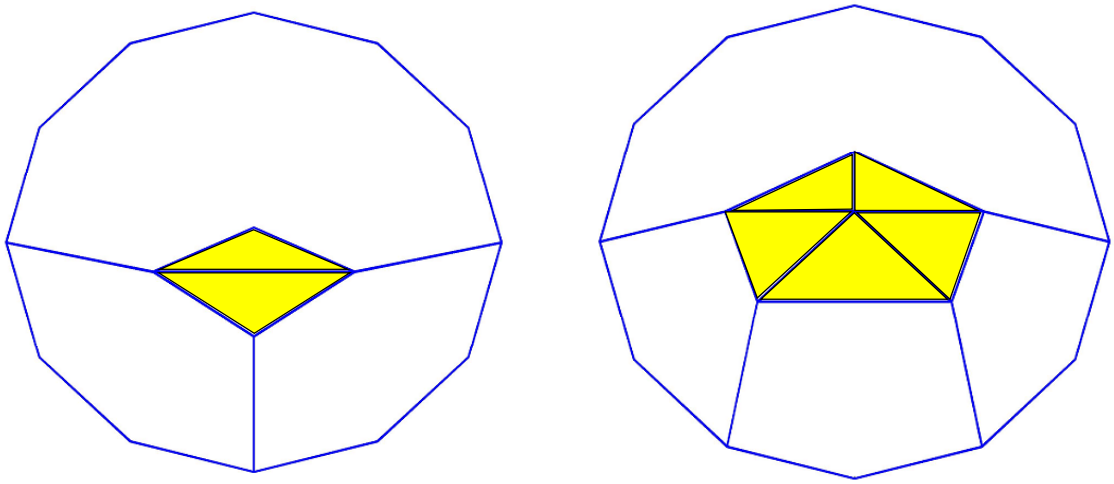
6.3 DESIGN OF A STEERING WHEEL

The objective of this research is to provide a skeleton based modeling tool with the ability to keep the correct topology for conceptual design. The thesis of this research is that skeletal modeling provides an effective way for designers to model the geometry of conceptual designs. The example presented in this section is meant to emulate such a scenario.

Assume that a designer has been given the task to design an automotive steering wheel, based on customer requests. The customer needs indicate that some customers like

to have more open spaces with which to grasp the steering wheel, while other customers like to grip the bottom of the steering wheel. The new design must meet safety constraints.

In this example, the designer generated two different designs. The first design, shown in Figure 6–13(a), has three spokes, while the second design (Figure 6–13(b)) has four spokes. Two skeletons were generated to model the concepts. The skeletons of these two designs contain important design information, such as geometry, functionality, etc. For example, the skeleton of design 1 reflects the design concept that allows the driver more options for holding the steering wheel on both sides. The skeleton of design 2 allows the driver to grip the bottom of the steering wheel, but the open spaces on sides are relatively small. Also, the topology of the skeletons, such as the open spaces between spokes, show the idea that spaces are needed for drivers to grip the steering wheel. These aspects of the skeletons reflect important features of the alternative designs that should be maintained when the solid models are generated.

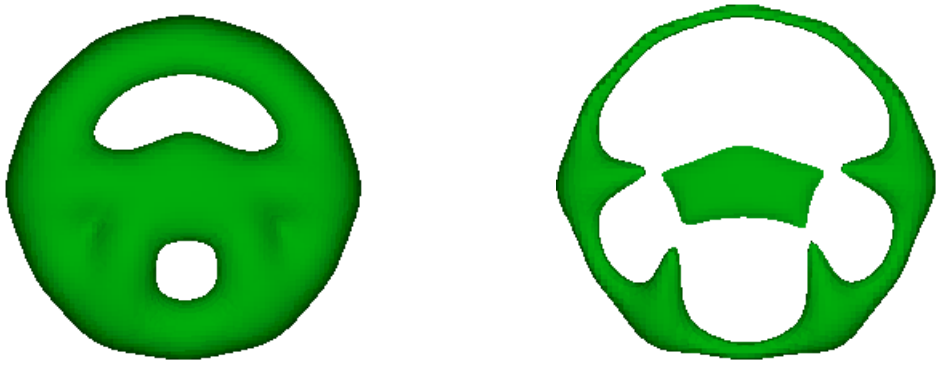


(a) Design 1 (b) Design 2

Figure 6–13. Skeletons of two steering wheel designs.

With these two skeletons specified, the next step was to visualize the alternatives so further design tasks can be proceed. In this stage, the existing topology of the skeletons must be kept, because the topology reflects the design intent.. After applying our for generating convolution surfaces and analyzing the topology of the surfaces with respect to the skeletons, the final shape of each design can be visualized by choosing a threshold value *T* within the recommended threshold intervals. The recommend threshold value *T* for design 1 is: [0.6230, 1.7890], and for design 2 is: [1.1235, 2.3363].

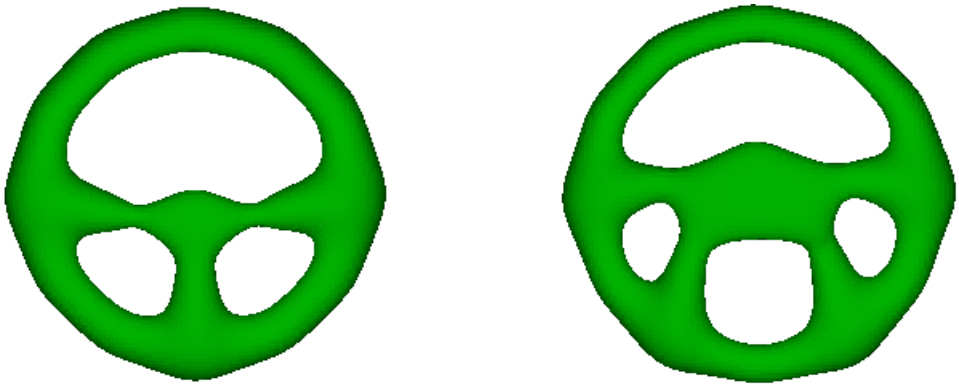
Without considering the topology issue, the related important information may be lost. For example, the final shape might fill in the spaces between the spokes, or with the connections between the spokes and the hub broken, which is not a steering wheel any more. These possibilities are illustrated in Figure 6–14 for design 2 with different threshold values that are out of the interval recommended by the system. In this particular case, the designer might be able to manually adjust the threshold to get the design he wants, but this might prove to be a time consuming trial and error process. Also, if the original skeleton is very complicated, for example, the skeleton of an airplane wing spar, human inspection is not reliable and robust. By integrating the methods presented in this research into a skeleton based modeling system, the quality of the generated surface can be guaranteed.



(a) Design 2: $T = 1.0$ (b) Design 2: $T = 2.4$

Figure 6–14. Steering wheel shapes that do not match the skeleton topology.

With topology analysis, as long as the threshold value T falls in the interval provided, the generated shape is guaranteed to have the same topology as that of the skeleton. Hence, the desired functionality, at least as it relates to geometry, can be assured. Examples of steering wheel shapes with different threshold values for the two designs are shown in Figure 6–15.



(a) Design 1 with $T = 1.5$. (b) Design 2 with $T = 1.8$

Figure 6–15. Final shapes of the two steering wheel designs.

The recommend value of T is an interval, so the final shapes can vary quite a bit (Figure 6–16). For example, when T = 1.2, the open spaces between the spokes are relatively small, and there may not be enough space for a driver to grip the steering wheel. When T = 2.3, the rim of the steering wheel is thin, and the connections between the spokes and the hub are weak. The designer can adjust the parameter values to improve the appearance of the design, and constraints can be added to further reduce the threshold value interval. The initial interval for T is a good starting point for exploring these parametric alternatives.

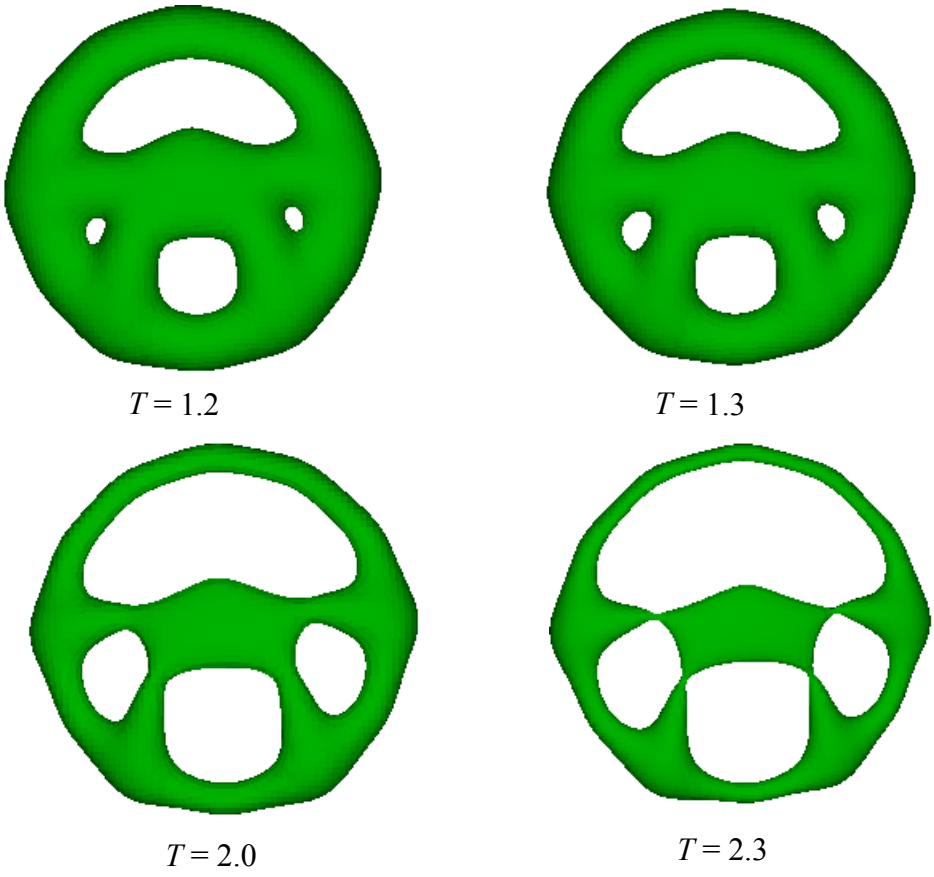


Figure 6–16. Different shapes for different values of T for design 2.

Once the desired shapes are generated, the concepts are ready for further analysis. For example, an early stress analysis of the design can be performed to determine, for instance, the required radius of the spokes. By choosing different T values from the initial interval, different designs with different spoke radii can be generated. In order to satisfy strength requirements, the interval may be further reduced. After embodiment, the final designs might resemble those in Figure 6–17.

(a) Design 1 (b) Design 2

Figure 6–17. Images of two steering wheel designs.

(a) http://www.rensup.com/Steering-Wheel-Covers/20575.htm
(b) http://priuschat.com/shop/product_info.php/products_id/7

This example illustrates the following:

1. A simple skeleton is a good way to generate geometry for conceptual design.
2. Even though the skeleton consists of simple geometry, it contains important design information, including the geometry and topology that define the design concept.

3. To further study a design concept, more detailed shape information can be generated for visualization or other analysis purposes. It is very important that the topology of the surface model match that of the skeleton so that important information that defines the design concept is not lost.
4. The initial interval of T is a very good starting point for design exploration. More detailed shapes generated with different T values can be used for stress analysis, cost estimation, weight estimation, etc.

6.4 COMPUTING TIME

The method described in this dissertation can find the CPs, but it takes a relatively long time and is not at speed needed to support interactive design. Table 6–3 shows the computing time for each of the cases described above.

Table 6–3. Time for computing CPs for different cases.

Example	Skeletal Primitives Used	Computing Time (s)
Bottle opener	9 line segments, 2 triangles	300
Steering wheel 1	15 line segments, 2 triangles	530
Steering Wheel 2	16 line segments, 5 triangles	720

When the number of skeletal elements increases, the computing time will also increase, since every skeletal element has a contribution to the composite surfaces. Also, since the formulas for line and triangle based convolution surfaces are complicated, the first gradient is more complicated. Many of the terms are repeated in the formula. They should be replaced by substitute variables to reduce the computing time.

6.5 SUMMARY

The examples shown in this chapter demonstrate that the proposed algorithms handle the topology problems properly. The ideal case occurs when the proper value of T can be chosen without changing the skeleton or its parameter values. However, many cases may require the skeleton or its parameters to be adjusted. A closed-form formula for generating convolution surfaces based on the tetrahedral element has not been found based on the Cauchy function as the kernel.

A tetrahedron can be simulated using four triangle elements and tracking the index-3 CP of the convolution surfaces generated by the four triangle elements. The threshold value T can be chosen so that no void is generated inside the four triangles.

A realistic conceptual design of a steering wheel is discussed by using the method developed in this research. The example illustrates the potential for skeletal modeling as a conceptual design tool.

Chapter 7 Conclusion and Future Work

In this dissertation we have argued that skeletal modeling is an approach to creating solid models in which the engineer designs with lower dimensional primitives such as points, lines, and triangles. The skeleton is then "*skinned over*" to create the surfaces of the three-dimensional object. Much skeletal modeling is limited to a single primitive – the point. Convolution surfaces can be generated from higher order skeletal elements (line segments and triangle elements, in addition to points) and can thus model more complex shapes. The resulting composite convolution surfaces are well-blended and smooth surfaces. However, it is possible for the topology of the convolution surfaces to change in a seemingly arbitrary way. We propose that the topology of the generated convolution surfaces should agree with that of the root skeleton.

This dissertation describes a way to control the topology of the generated surfaces. The proposed method is based on analyzing the topology of the surfaces using Morse theory. Critical points (CPs) of the convolution surfaces are identified by searching the space bounded by the convex hull of the entire skeleton. CPs provide valuable information about the topology of the convolution surfaces. The CP type indicates what kind of topology change occurs as the threshold value T reaches the critical value at the CP. The CP location indicates where the topology change occurs. The topology of the skeleton can be determined readily from an explicit representation of the connectivity of the skeletal elements.

By comparing the Betti numbers of the skeleton and composite convolution surfaces, we can determine whether the two match topologically. The Betti numbers

indicate on a *global* level whether the skeleton and convolution surfaces match topologically. Local topology agreement is tested by comparing the geometry of the loops associated with index-2 and index-3 CPs. If the local topology matches, the proper threshold value T is provided to the user to generate the surfaces. If no proper T value can be found, the parameters of the convolution surfaces can be adjusted. A strategy for adjusting the skeleton as well as the property values of the convolution surfaces has been developed. The width of the kernel function, a parameter s, is adjusted based on certain rules to alter the topology of the convolution surfaces in cases where adjustment of the threshold T does not produce a match.

7.1 CONTRIBUTIONS

The contributions of this research include:

- Use of convolution surfaces as a modeling tool to generate smooth surfaces for a skeletal modeling system for conceptual design. Currently the analytic solutions for points, line segments, curves, arcs, triangles, and planes are known. An approach to simulating tetrahedral elements is also discussed.
- A method for analyzing the topology of convolution surfaces with geometric primitives other than points. Previous research reports only topology analysis for point based implicit surfaces. Morse theory is applied to analyze the topology of the convolution surfaces.
- An effective algorithm for identifying the critical points of convolution surfaces by analyzing the geometry of the underlying skeleton. We propose that the skeleton represents not only the final shape of the part, but also the desired topology of the part. By partitioning the skeleton into convex subregions, the CPs can be identified.

- A strategy to match the topology of convolution surfaces with the root skeleton. The shape generated by the convolution surfaces is controlled by adjusting appropriate parameters to maintain topological agreement during manipulation of the skeleton.

7.2 FUTURE WORK

While the research has met the intended objectives, the approach developed has limitations. These include:

- Convolution surfaces: The shape of the convolution surfaces can be difficult to control. For convolution surfaces based on the Cauchy kernel function, aside from the skeletal geometry, there are only two parameters that control the shape. The geometry has a significant influence on the final shape. However, certain threshold values may not generate any shape at all. This may frustrate the user.
- The cross sections of the objects generated by convolution surfaces are typical circular. This is suitable to generate branch shaped objects, such as the manifold illustrated in Chapter 1. However, real objects often flat regions or sharp corners. Convolution surfaces alone cannot model such features.
- The generated surfaces cannot be imported into commercial CAD system yet. A deployed system must be able to convert the generated surfaces to other data formats, such as boundary representations (B-Rep), so that the generated model can be directly used for detailed design.
- Topology analysis: We have developed a method for identifying the CPs of convolution surfaces. The finds all critical points for the convolution surfaces. But the performance to date is not at interactive speeds. More efficient search algorithms are needed for calculating the CPs.

- Currently, only 2D skeleton shapes are partitioned. The method should be extended to 3D skeletons, since real objects are three dimensional.
- Morse theory can handle only non-degenerate CPs. CPs become degenerate when skeletal elements are placed far away from each other. In this research, we only consider cases where the CP is non-degenerated.
- While we have developed reliable heuristics to guide the search for non-degenerate CPs, we have no proof that we can locate all CPs in every situation. Further, we have no method to determine an upper bound on the number of CPs *a priori*.

Based on above limitations of this research, suggestions for future research are included here.

- Non-circular cross section

Normally, for a line segment skeletal element, the convolution surface generated is a cylinder. The cross-section is circular. However, for more general cross sections, the radius is not constant. The radius of the cylinder can be changed by multiplying the field function with a profile function along the length of the line segments [Sher98]. Based on this idea, different cross-section forms can be obtained by multiplying the field function with a cross-section function. An example is shown in Figure 7–1, where the cross section is an arbitrary shape. The base shape is a cylinder formed by a line segment and the original cross section is circular. In the local coordinate system, the circle area is cut by three curves (Figure 7–2):

$$\begin{cases} xy = c; & x < 0 \,\&\, y < 0; \quad 0 < c < R \\ x + y = b; & 0 < b < \sqrt{2}R \\ x = a; & 0 < a < R \end{cases} \tag{7–1}$$

where *R* is the radius of the cylinder, and *x* and *y* are local coordinates. The yellow area is the final cross section of the convolution surfaces. Non-circular cross section objects should be thoroughly studied flexibility for designers.

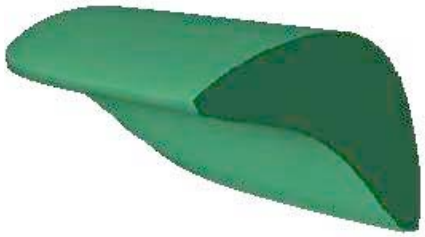


Figure 7–1. Non-circular cross section shape.

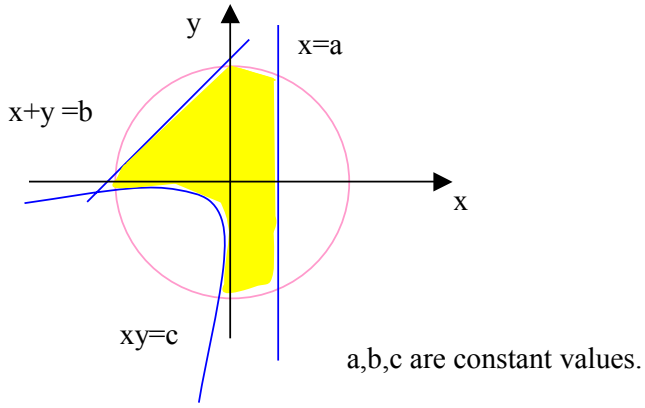


Figure 7–2. Detail of cross section of shape in Figure 7–1.

- Converting to B-Rep

The system should be able to convert the model generated by convolution surfaces to a B-Rep so that it can be imported into the commercial CAD systems for detailed design. The B-Rep can be obtained from the representationally simpler implicit

surface polygonization. Either ordered sequences of edges around each vertex or ordered lists of edges around each face are sufficient to reproduce an adjacency graph that is embedded in a two dimensional manifold. In other words, the points and polygons that are produced by conventional polygonizers are sufficient to generate a B-Rep [Bloo97b].

- Partitioning 3D spaces

To provide designers with the ability to specify 3D skeletons, partitioning the skeleton in 3D is necessary. A strategy is needed for portioning a 3D skeleton into convex subregions. Several researchers report work on partitioning 3D polyhedra [LA06, BD92, CDST95]. Lien and Amato [LA06] use a method to approximate convex decomposition of polyhedra. Concave vertices are considered and removed based on their "importance" to the object, and tolerance value is used to determine whether a vertex is concave. This approach may have other benefits for our modeler. We have found that many subregions did not contain index-2 CPs, but the algorithm still searches for them. The minimum distance from a concave vertex to other skeletal element might be a way to measure the importance of a concave vertex.

- Localizing computation of convolution surfaces

In the current implementation, once the parameters are adjusted, the topology of all convolution surfaces is recomputed. To improve computational performance, an algorithm should to be developed to handle local topology changes instead of regenerating unchanged convolution surfaces.

- Degenerate critical points

For degenerate CPs, Morse theory cannot be applied. Catastrophe theory [Gilm93, CS93] appears to provide a solution. The main idea for removing degenerate CPs is to add a small perturbation at the degenerate CP location. Thorough study of the Catastrophe theory and its applicability to this problem should be conducted.

- Upper bound on CPs

The search for CPs in this research is based on heuristics, and an upper bound on the number of CPs is not known. This is a critical area of study that is needed to ensure that no CPs are neglected.

- Accelerating the search for CPs

Computing CPs is very time consuming, especially for higher dimensional skeletal elements. An accelerated algorithm should be developed to run at interactive speeds when the geometry of skeleton and/or the parameter values of the corresponding convolution surfaces are modified. Reducing the number of convex subregions and using symmetry information from the skeleton might reduce the computational time.

- Integrated system

The proposed algorithms provide the basis for a conceptual design system. A robust implementation and integration of these algorithms is needed to provide a usable tool for conceptual design.

7.3 APPLICATIONS

- Mechanical Conceptual Design

This research describes a skeleton-based mechanical conceptual design tool. With it, the user can design shapes using simple lower dimension skeletal elements. The final shape consists of more complicated 3D surfaces. The approach provides a model with smooth surfaces that have the correct topology, ensuring that those aspects of the desired functionality of the initial design that are related to topology are represented.

- Initial cost estimation

After generating the conceptual model (final shape), conceptual process planning can be performed. Conceptual process planning is an activity of preliminary manufacturability assessment and cost estimation of conceptual design. Conceptual

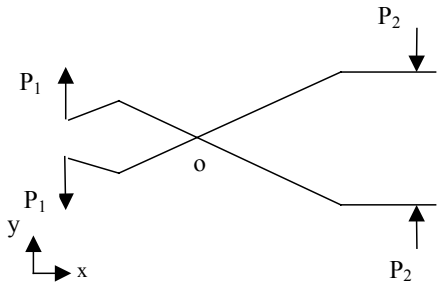
process planning enables product developers: (1) to assess manufacturability by determining and selecting available manufacturing processes and manufacturing resources; (2) to estimate manufacturing cost for affordability assessment; and (3) to reduce product development time by preparing production activities and acquiring necessary manufacturing resources as the concept of a product is sketched and the critical parameters of the conceptual product are defined [FZ99]. Manufacturability analysis can be conducted as early as the skeleton design stage. For example, if the skeleton is designed to have a void inside, traditional manufacturing methods cannot be used.

- Strength analysis for conceptual design

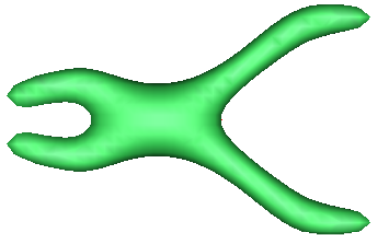
Mechanical components typically have strength requirements. During conceptual design stage, the skeleton can serve as an abstract model of the final component. The strength of the mechanical component can be analyzed based on the skeleton, with the addition of material and cross-sectional geometry information. This approach promises to reduce the computing time by simplifying the model.

Based on the desired loads applied to the component, the loads applied to each skeleton element can be calculated. Consider the simple example of a clamp, shown in Figure 7–3. The load is added through the ends of the clamp. Assume that the load is uniform. To simplify the analysis, we will analyze one part of the clamp, and we will consider this component to be a straight beam. Since in the conceptual design stage, most information is incomplete and uncertain, we can make assumptions about the parameters of the skeletal elements. In this case, we assume that the clamp is subjected to pure bending. The loads P_1 and P_2 are applied to the clamp, and σ_{max} is the desired maximum stress. For a circular cross section, $I = \pi R^4 / 4$. With the generated convolution surfaces, the radius can be calculated. The bending stress is calculated as $\sigma = \frac{M_{\max}}{I} = \frac{P_1 L_1}{I}$, where I is the moment of inertia, P_1 is the force applied to the left end, and L_1 is the distance

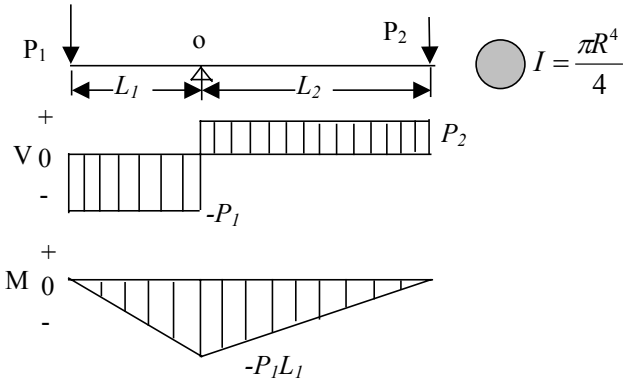
from the left end to joint *O*. Next the designer can compare σ and σ_{max} to decide whether the properties of the convolution surfaces need to change.



(a) clamp skeleton



(b) clamp convolution surface



(c) force analysis (simplified model).

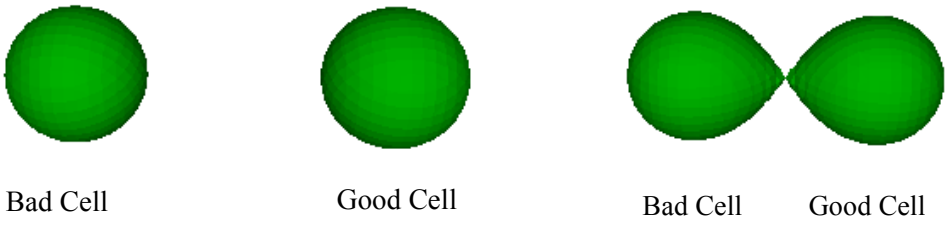
Figure 7–3. Stress analysis of a conceptual clamp design.

- Motion simulation

When modeling a moving component, kinematic simulation using a complete 3D surface model may be more complicated than necessary. The skeleton, a lower dimensional object, provides an alternative method for motion studies, as the locations of the skeletal elements can be readily tracked. The corresponding convolution surfaces can then be generated based on each configuration of the skeleton. This is the classical method of kinematic design of linkages.

- Collision detection

Collision detection between objects is not completely solved. For example, there are open problems for detecting collisions between blood cells, particularly between bad blood cells that carry disease. When bad blood cells move, they need to be monitored to determine whether they touch other cells, which might cause infection in other good blood cells. To detect collisions between cells, convolution surfaces can be used to model the blood cells. When two blood cells contact each other, the topology between the two cells changes from two single objects to one connected object. If the topology changes of the convolution surfaces are monitored, potential collisions can be controlled. Computing speed is critical for this application, since the blood cells are moving constantly. However, once we know the location of the blood cells, the problem becomes solvable. Convolution surfaces are determined by their relative locations, as well as the geometry of the blood cells (skeleton). The formula for the point primitive is much simpler than the formulas for line segments or triangle elements, and the computing time is greatly reduced.



(a) no interaction (b) starting to interact

Figure 7–4. Use of convolution surfaces to model blood cells.

The index-1 CPs can be closely monitored to detect any topology changes between the cells, which would indicate a collision. When two cells are close to each other, an index-1 critical point will appear between them. In this way, the collision can be detected if we assume that the blood cells do not change in size.

Appendix A Convolution Surface Field Function [Shre98]

The field function is the integral of formula (3–3). For convenience, it is listed as equation A-1:

$$f(\vec{p}) = g(\vec{p}) \otimes h(\vec{p}) = \int_{R^3} g(\vec{r}) h(\vec{p} - \vec{r}) d\vec{r} \ . \qquad \text{(A-1)}$$

Different geometric primitives have different field functions. The field functions for a number of primitives can be derived, including:

- Point sources
- Line segments
- Arcs
- Triangles
- Planes

These field functions for different primitives with the Cauchy kernel are described in detail below.

A.1 POINT SOURCES

A point is a 0-dimensional object, so its geometry function is a simple delta-function at the point $\vec{p}$. Integration of equation A-1 yields the field function, projected into 3D space as:

$$f_{point}(\vec{r}) = \frac{1}{(1 + s^2 |\vec{p} - \vec{r}|^2)^2}, \qquad \text{(A–2)}$$

where $\vec{r}$ is a point of interest.

A.2 LINE SEGMENTS

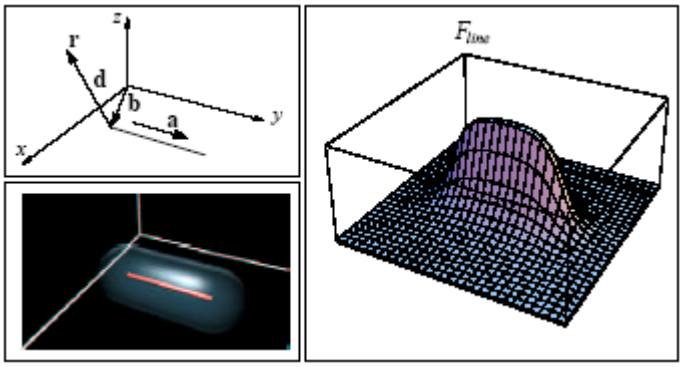


Figure A–1. Line primitive [Sher98].

A line segment of length l can be specified as:

$$\vec{p}(t) = \vec{b} + t\vec{a},\ 0 \le t \le l,$$

where $\vec{b}$ is the base point and $\vec{a}$ is the normalized direction of the line. The squared distance from an arbitrary point $\vec{r}$ and a point on the line segment is:

$$r^2(t) = d^2 + t^2 + 2t\vec{d} \bullet \vec{a},$$

where $d = \left\|\vec{d}\right\|$ is the magnitude of a vector from the segment base to $\vec{r}$: $\vec{d} = \vec{r} - \vec{b}$. After integration, the field function for line segments is:

$$\begin{aligned} f_{line}(\vec{r}) &= \int_0^l \frac{dt}{(1 + s^2 r^2(t))^2} \\ &= \frac{h}{2p^2(p^2 + s^2h^2)} + \frac{l-h}{2p^2q^2} + \frac{1}{2sp^3}\left[\operatorname{atan}\left(\frac{sh}{p}\right) + \operatorname{atan}\left(\frac{s(l-h)}{p}\right)\right], \end{aligned} \tag{A–3}$$

where $h = \vec{d} \bullet \vec{a}$, and p and q are distance terms, given by:

$$p^2 = 1 + s^2(d^2 - h^2),$$
$$q^2 = 1 + s^2(d^2 + l^2 - 2lh).$$

A.3 ARCS

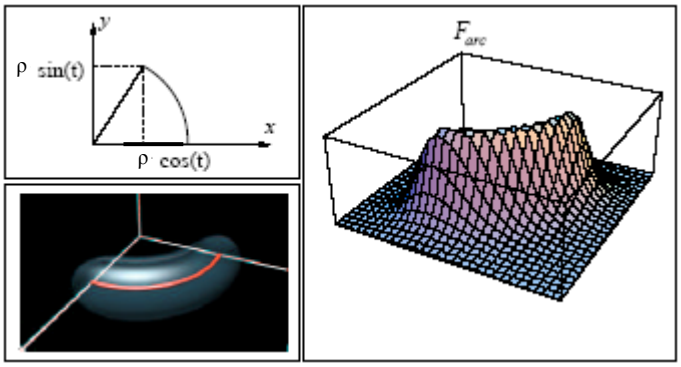


Figure A–2. Arc primitive [Sher98].

An arc and its distance function $r^2(t)$ are conveniently defined in the arc's local z-aligned coordinate system as:

$$\vec{p}(t) = (\rho\cos(t), \rho\sin(t), 0), \quad 0 \le t \le \theta,$$
$$r^2(t) = (x - \rho\cos(t))^2 + (y - \rho\sin(t))^2 + z^2,$$

where $\vec{p}$ is the point on the arc, *x, y,* and *z* are the coordinates of the point of interest, ρ is the radius of the circle the arc lies on, θ is the arc angle, and t is the parameter. Integrating with respect to the angle, the field function in local coordinates is:

$$
\begin{aligned}
f_{arc}(x,y,z) &= \int_0^{\theta} \frac{dt}{(1+s^2r^2(t))^2} \\
&= \frac{by}{xp^2(kx-b)} + \frac{k(x^2+y^2)\sin(\theta)-by}{xp^2(k(x\cos(\theta)+y\sin(\theta))-b)} \\
&+ \frac{2b}{p^3}(\text{atanh}[\frac{ky}{p}] + \text{atanh}[\frac{(kx+b)\tan(\theta/2)-ky}{p}]),
\end{aligned}
\tag{A–4}
$$

where $k = 2\rho s^2$ and distance terms *d, b,* and *p* are:

$$
\begin{aligned}
d^2 &= x^2 + y^2 + z^2, \\
b &= 1 + \rho^2 s^2 + s^2 d^2, \\
p^2 &= -\rho^4 s^4 + 2\rho^2 s^2 (s^2(d^2 - 2z^2) - 1) - (1 + s^2 d^2)^2.
\end{aligned}
$$

A.4 TRIANGLES

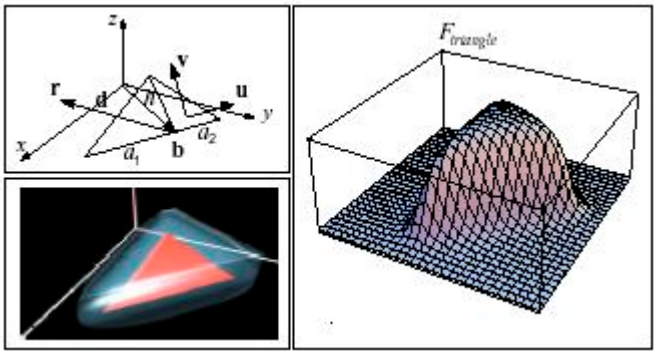


Figure A–3. Triangle primitive [Sher98].

A triangle primitive requires integration in two dimensions. First a triangle is split along the longest edge into two right triangles, with base lengths of a_1 and a_2 respectively as shown in Figure A–3. Next the field function for the right half is obtained:

$$f_{right} = \int_{y=0}^{h-\frac{hx}{a_2}} \int_{x=0}^{a_2} \frac{dxdy}{(1+s^2r^2(x,y))^2}. \tag{A–5}$$

Next, f_{left} is derived from f_{right} by replacing *x* with –*x* and a_2 with a_1. Finally the field function is the summation of f_{left} and f_{right}.

The following parameters are defined: point $\vec{b}$, the projection onto the longest edge of the opposite vertex; vectors $\vec{u}$ and $\vec{v}$ that form the local surface coordinate system, with $\vec{b}$ as its origin and aligned in the direction of the longest edge; and *h*, the distance from $\vec{b}$ to the apex of the triangle. Introducing the vector $\vec{d} = \vec{r} - \vec{b}$, where $\vec{r}$ is the point of interest, and scalars $u = \vec{d} \bullet \vec{u}$, $v = \vec{d} \bullet \vec{v}$, the field function is:

$$\begin{aligned} f_{triangle}(\vec{r}) = \frac{1}{2qs}\Big(&\frac{n}{A}\arctan\Big[\frac{vh + a_1(a_1+u)}{A}\Big] + \arctan\Big[\frac{gh + a_1u}{-A}\Big]\Big) \\ &+ \frac{m}{B}\Big(\arctan\Big[\frac{vh + a_2(a_2-u)}{A}\Big] + \arctan\Big[\frac{gh - a_2u}{B}\Big]\Big) \\ &+ \frac{v}{C}\Big(\arctan\Big[\frac{a_1+u}{C}\Big] + \arctan\Big[\frac{a_2-u}{C}\Big]\Big)\Big), \end{aligned} \tag{A–6}$$

where:

$$\begin{aligned} A^2 &= a_1{}^2w + h^2(q+u^2) - 2a_1hug, \\ B^2 &= a_2{}^2w + h^2(q+u^2) + 2a_2hug, \\ C^2 &= 1/s^2 + d^2 - u^2, \\ g &= v - h, \\ q &= C^2 - v^2, \\ w &= C^2 - 2vh + h^2, \\ m &= a_2g + uh, \\ n &= uh - a_1g. \end{aligned}$$

A.5 PLANE

The field function for an unbounded plane primitive is computed as follows:

$$f_{plane}(\vec{r}) = \int_S \frac{dS}{(1+s^2r^2)^2} = \int_0^\infty \frac{2\pi t dt}{(1+s^2r^2(t))^2} = \frac{\pi}{s^2(1+s^2d^2)}. \qquad \text{(A–7)}$$

where d^2 is the distance between point $\vec{r}$ and the plane and dS is the area of an integration ring.

Bibliography

[AGJ00] U. Adamy, J. Giesen, and M. John, "New Techniques for Topologically Correct Surface reconstruction", *Proceeding of IEEE Visualization*, pp. 373-380, 2000.

[AC98]] E. Akleman and J. Chen, "Guaranteeing 2-manifold Property for Meshes", In *Proceedings of the International Conference on Shape Modeling and Applications*, 1998.

[ACS00] E. Akleman, J. Chen, and V. Srinivasan, "A New Paradigm for Changing Topology of 2- Manifold Polygonal Meshes", In *Pacific Graphics'2000*, 2000.

[AB99] N. Amenta and M. Bern, "Surface Reconstruction by Voronoi Filtering", *Discreet Computational Geometry,* Vol. 22, pp. 481-504, 1999.

[ACDL00] N. Amenta, S. Choi, T. Dey, and N. Leekha, "A Simple Algorithm for Homeomorphic Surface Reconstruction", *ACM Symposium on Computational Geometry*, pp. 213-222, 2000.

[ACK01] N. Amenta, S. Choi, and R.K. Kolluri, "The Power Crust", *Proceeding of the Sixth ACM Symposium on Solid Modeling and Applications*, ACM Press, New York, pp. 249-266, 2001.

[AJC02] A. Angelidis, P. Jepp, and M. P. Cani, "Implicit Modeling with Skeleton Curves: Controlled Blending in Contact Situations", *Shape Modeling International 2002*, Banff, Canada, May 17-22, 2002.

[ABE07] D. Attali, J.D. Boissonnat, and H. Edelbrunner, "Stability and Computation of Medial Axes – a State-of-the-Art Report", *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, Springer-Verlag, Mathematics and Visualization, 2007.

[Axen99] U. Axen, "Computing Morse Functions on Triangulated Manifolds", in *Proceedings of the SIAM Symposium on Discrete Algorithms (SODA),* 1999.

[AE98] U. Axen and H. Edelsbrunner. "Auditory Morse Analysis of Triangulated Manifolds", *Mathematical Visualization*, H.C. Hege and K. Polthier, Eds., Springer-Verlag, Heidelberg, pp. 223-236, 1998.

[BBX97] C. Bajaj, F. Bernardini, and G. Xu, "Reconstruction of Surfaces and Functions-on-surfaces from Unorganized Weighted Points", *Algorithmica*, Vol. 19, pp. 243-261, 1997.

[BCL96] C. Bajaj, E. Coyle, and K. Lin, "Arbitrary Topology Shape Reconstruction From Planar Cross Sections", *Graphical Models and Image Processing*, Vol. 58, No. 6, pp. 524-543, 1996.

[BD92] C. Bajaj and T. K. Dey. "Convex Decomposition of Polyhedra and Robustness". *SIAM J. Comput.*, Vol. 21, pp. 339-364, 1992.

[BS98] C. Bajaj and D. Schikore, "Topology Preserving Data Simplification with Error Bounds", *Computer and Graphics*, Vol. 22, No. 1, pp. 3-12, 1998.

[BBCS99] F. Bernardini, C. Bajaj, J. Chen, and D. Schikore, "Automatic Reconstruction of 3D CAD Models from Digital Scans", *International Journal on Computer Geometry and Application,* Vol. 9, No. 4-5, pp. 327-369, 1999.

[BK02] S. Bischoff and L. Kobbelt, "Isosurface Reconstruction with Topology Control", *Proceeding of Graphics*, Vol. 3, No. 10, pp. 246-255, 2002.

[Blin82] J.F. Blinn, "A Generalisation of Algebraic Surface Drawing", *ACM Transactions on Graphics*, Vol. 1, No. 3, pp. 135-256, 1982.

[Bloo95] J. Bloomenthal, "Skeleton Design of Natural Forms", *Ph.D. dissertation, The university of Calgary,* 1995.

[Bloo97a] J. Bloomenthal, "Bulge Elimination in Convolution Surfaces", *Computer Graph Forum*, Vol. 16, No. 1, pp. 31-41, 1997.

[Bloo97b] J. Bloomenthal (Ed.), *Introduction to Implicit Surfaces*, Morgan Kaufmann, ISBN 1-55860-233-X, 1997.

[BS91] J. Bloomenthal and K. Shoemake, "Convolution Surfaces", *SIGGRAPH '91 conference proceedings of Computer graphics annual conference series*, pp. 251-256, 1991.

[BCV03] J. D. Boissonnat, D. Cohen-Steiner, and G. Vegter, "Meshing Implicit Surfaces with Certified Topology", *ECG technical Report NO.: ECG-TR-244104-03*, 2003.

[BNV96] A. Bottino, W. Nuij, and K. Van Overveld, "How to Shrinkwrap Through a Critical Point: and Algorithm for the Adaptive Triangulation of Iso-surfaces with Arbitary Topology", *Proceedings on Implicit surfaces '96*, pp. 53-72. Oct. 1996.

[Buch02] R. O. Buchal, "Sketching and Computer-Aided Conceptual Design", *Proceedings of the 7th International Conference on Computer Supported Cooperative Work in Design*, pp. 112-119, Sep. 25-27, 2002.

[Cani93] M. P. Cani-Gascuel, "An Implicit Formulation for Precise Contact Modeling Between Flexible Solids", In *Computer Graphics (Annual Conference Series.)*, pp. 313-320, 1993.

[CH01] M. P. Cani and S. Hornus, "Subdivision Curve Primitives: a New Solution for Interactive Implicit Modeling", In *Shape Modelling International*, Italy, May 2001.

[CS93] D. P. Castrigiano, and S. A. Sayes, "*Catastrophe Theory*", Addison-Wesley Publishing Company, 1993.

[Chaz81] B. Chazelle, "Convex Decompositions of Polyhedra", In *Proceeding of 13th Annual ACM Symposium on Theory of Computing*, pp. 70-79, 1981.

[CDST95] B. Chazelle, D. P. Dobkin, N. Shouraboura, and A. Tal. "Strategies for Polyhedral Surface Decomposition: An Experimental Study", In *Proceeding of 11th Annual ACM Symposium of Computational Geometry,* pp. 297-305, 1995.

[Cheng89] P. Cheng, "Using Plane Vector Fields to Obtain All the Intersection Curves of Two General Surfaces", *Theory and Practice of Geometric Modeling,* New York, 1989.

[CBS96] B. Crespin, C. Blanc, C. Schlick, "Implicit Sweeps Objects*", Proceeding of Eurographics'96*, Paitiers, France, 1996.

[dBvKOS97] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *"Computational Geometry Algorithms and Applications",* Springer - Verlag, Berlin Heidelberg, 1997.

[DE95] C. J. A. Delfinado and H. Edelsbrunner, "An Incremental Algorithm for Betti Numbers of Simplicial Complexes on the Sphere", *Computer Aided Geometry Design*, No. 12, pp. 771-784, 1995.

[DH94] T. Delmarcelle and L. Hesselink, "The Topology of Symmetric, Second-order Tensor Fields", *Proceedings IEEE Visualization '94,* pp. 140-147, October 1994.

[DES99] T. K. Dey, H. Edelsbrunner and S. Guha, "Computational Topology", *Advances in Discrete and Computational Geometry*, pp. 109-143, eds.: B. Chazelle, J. E. Goodman and R. Pollack, Contemporary Mathematics 223, AMS, Providence, 1999.

[Edd77] W. F. Eddy, "A New Convex Hull Algorithm for Planar Sets", *ACM Transaction on Mathematics Software*, Vol. 3, pp. 398-403 and 411-412, 1977.

[Edel92] H. Edelsbrunner, "Weighted Alpha Shape", *Technical Report UIUCDCS-R-92-1760*, Department of Computer Science, University of Illinois at Urbana, IL, 1992.

[Edel98] H. Edelsbrunner, "Shape Reconstruction with Delaunay Complex", *In Proceeding of Symposium of Latin American Theoretical Informatics, 1998*, Campinas, Brazil, Springer-Verlag Lecture Notes 1380, pp. 119-132, 1998.

[Edel00] H. Edelsbrunner, "Triangulations and Meshes in Computational Geometry", *Acta Numerica*, pp. 133-213, 2000.

[EKS83] H. Edelsbrunner, D. G. Kirkpartrick, and R. Seidel, "On the Shape of a Set of Points in the Plane", *IEEE Transactions on Information Theory*, Vol. 29, No. 4, pp. 551-559, 1983.

[ELZ02] H. Edelsbrunner, D. Letscher, and A. Zomorodian, "Topological Persistence and Simplification", *Discrete Computational Geometry*, Vol. 28, pp. 511-533, 2002.

[EM94] H. Edelsbrunner and E.P. Mucke, "Three-dimensional Alpha Shapes", *ACM Transactions on Graphics*, Vol. 13, No. 1, pp. 43-72, 1994.

[ESV98] J. El-Sana and A. Varshney, "Topology Simplification for Polygonal Virtual Environments", *IEEE Transactions on Visualization and Computer Graphics*, Vol. 4, No. 2, pp. 133-144, June 1998.

[Fer97] E. Ferley, M. C. Gascuel, and D. Attali, "Skeletal Reconstruction of Branching Shapes", *Computer Graphics Forum*, Vol. 16, No. 5, December 1997.

[FG01] P. A. Firby, and C. F. Gardiner, "*Surface Topology",* 3rd Edition, *Howood Publishing*, Chichester, W Sussex, 2001.

[GG99] T. W. Gamelin, R. E. Greene, "*Introduction to Topology*", 2nd Edition, Dover Publications, Inc., Mineola, New York, 1999.

[GD95] S. M. Gelston and D. Dutta, "Boundary Surface Recovery From Skeleton Curves and Surfaces**",** *Computer Aided Geometric Design*, Vol. 12, No. 1, pp. 27-51, 1995.

[GP00] T. Gerstner and R. Pajarola, "Topology Preserving and Controlled Topology Simplifying Multiresolution Isosurface Extraction", In *Proceedings of IEEE Visualization 2000*, pp. 259-266, 2000.

[Gilm93] R. Gilmore, "*Catastrophe Theory for Scientists and Engineers*", Dover Publications, 1993.

[Gree83] Daniel H. Greene, "The Decomposition of Polygons into Convex Parts", *In Advanced Computing Research: Computational Geometry, Franco P. Preparata, editor,* Vol. 1, pp. 235-259. JAI Press, Greenwich, Conn., 1983.

[Gosh05] A. Goshtasby, "Plus Curves and Surfaces", *The Visual Computer*, Vol. 21, pp. 4-16, 2005.

[GW01] I. Guskov and Z. Wood, "Topological Noise Removal", *Proceedings of Graphics Interface*, 2001.

[GW95] A. Guy and B. Wyvill. "Controlled Blending for Implicit Surfaces Using a Graph". In *Proceeding of Implicit Surfaces '95. Eurographics*, pp. 107-112. 1995.

[Hart98] J. C. Hart, "Morse Theory for Implicit Surface Modeling", *Mathematical Visualization*, pp. 257-268, Oct. 1998.

[Hart99] J. C. Hart, "Computational Topology for Shape Modeling", *International conference on shape modeling and applications*, Aizu-Wakamatsu, Japan, March 01-04, 1999.

[HH91] J. L. Helman and L. Hesselink, "Visualizing Vector Field Topology in Fluid Flows", *IEEE Computer Graphics and Applications*, pp. 36-46, May 1991.

[HM85] S. Hertel and K. Mehlhorn, "Fast Triangulation of the Plane with Respect to Simple Polygons", *Information & Control*, Vol. 64, No. 1-3, 1985.

[HDDMS92] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface Reconstruction from Unorganized Points", *ACM SIGGRAPH 1992*, pp. 71-78, 1992.

[HAC03] S. Hornus, A. Angelidis, and M. P. Cani, " Implicit Modeling Using Subdivision Curves", *The Visual Computer,* Vol. 19, No. 2-3, pp. 94-104, 2003.

[JTFP01] X. Jin, C. L. Tai, J. Feng, and Q. Peng, "Convolution Surfaces for Line Skeletons with Polynomial Weight Distributions. *Journal of Graphics Tools*, Vol. 6, No. 3, pp.17-28, 2001.

[KW91] Z. Kacic-Alesic and B. Wyvill, "Controlled Blending of Procedural Implicit Surfaces", In *Proceeding* of *Graphics Interface '91*, pp. 236-245. 1991.

[KS85] M. Keil and J. R. Sack, "Computational Geometry, Chapter Minimum Decomposition of Geometric Objects", pp. 197-216, *North-Holland*, 1985.

[Keil00] J. M. Keil, “Polygon Decomposition”, *Handbook of Computational Geometry*, ed. J. R. Sack and J. Urrutia, Elsevier Science Publishing, 2000.

[LA04] J. M. Lien and N. M. Amato, “Approximate Convex Decomposition of Polygons”, *Proceeding of 20th Annual ACM Symposium on Computational Geometry*, Brooklyn, New York, USA, June 8-11, 2004.

[LA06] J. M. Lien and N. M. Amato, “Approximate Convex Decomposition of Polyhedra”, *Technical Report, TR06-002*, Parasol Laboratory, Department of Computer Science, Texas A&M University, 2006.

[LB03] A. Lopes and K. Brodlie, “Improving the Robustness and Accuracy of the Marching Cubes Algorithm for Isosurfacing”, *IEEE Transaction on Visualization and Computer Graphics.* Vol. 9, No. 1, 2003.

[MJFP01] T. Ma, X. Jin, J. Feng, and Q. Peng, "Convolution Surface Modeling Based Planar NURBS Curve Skeletons”, *Proceedings of CAD & Graphics'2001*, International Academic Publishers, August 22-24, 2001, Kuming, China, pp.148-154, 2001.

[MS98] J. MaCormack and A. Sherstyuk, “Creating and Rendering Convolution Surfaces”, *Computer Graphics forum*, 1998.

[MP02] N. O. J. Malcolm and P. L. A. Popelier, “An Improved Algorithm to Locate Critical Points in a 3D Scalar Field as Implemented in the Program MORPHY”, *Journal of Computational Chemistry*, Vol. 24, No. 4, 2003.

[MYB00] K. Abdel-Malek, J. Yang, and D. Blackmore, "Closed-form Swept Volume of Implicit Surfaces", *Proceedings of DETC’00, 2000 ASME Design Automation Conference*, Baltimore, MD, September 10-13, 2000.

[MAR00] J. L. Mari and J. Sequeira, “Using Implicit Surfaces to Characterize Shapes Within Digital Volumes”, *RECPAD 2000 Proceedings*, Porto, Portugal, May 2000, pp. 285-289, 2000.

[MS01] J. L. Mari and J. Sequeira, “Reconstruction by Implicit Using Discrete and Continuous Skeletons”, *Hermes, Revue Internationale d’Informatique Graphique et de CFAO*, Vol. 16, No. 2 (Tridimensional reconstruction), pp. 203-212, 2001.

[MS04] J. L. Mari and J. Sequeira, “Closed Free-Form Surface Geometrical Modeling – A New Approach with Global and Local Characterization”, *International Journal of Image and Graphics (IJIG)*, Vol. 4, No. 2, pp. 241-262, World Scientific Publishing, April, 2004.

[Mars87] K. M. Marshek, “Design of Machine and Structural Parts”, *Wiley-Inter Science Publication, John Wiley & Sons*, 1987.

[Miln63] J. Milnor, "Morse Theory", *Annals of Mathematics Studies*, Vol. 51, Princeton University Press, NJ, 1963.

[NGH04] X. Ni, M. Garland, and J. C. Hart "Fair Morse Functions for Extracting the Topological Structure of a Surface Mesh", *Proceeding of Siggraph 2004*, 2004.

[NPK06] T. Nieda, A. Pasko, and T. L. Kunii, "Detection and Classification of Topological Evolution for Linear Metamorphosis", *Visual Computing*, Vol. 22, pp. 346-356, 2006.

[NB93] P. Ning and J. Bloomenthal, "An Evaluation of Implicit Surface Tilers*", Computer Graphics and Applications,* Vol. 13, No. 6, pp. 33-41, Nov. 1993.

[NHKSO85] H. Nishimura, M. Hirai, T. Kawai, I. Shirakawa, and K. Omura, "Object Modeling by Distribution Function and a Method of Image Generation", *The Transactions of the Institute of Electronics and Communication Engineers of Japan*, Vol. J68-D, No. 4, pp. 718-725, in Japanese, translated into English by Takao Fujuwara, 1985.

[OP04] S. Oeltze and B. Preim, "Visualization of Anatomic Tree Structures with Convolution Surfaces", *IEEE TCVG Symposium on Visualization*, 2004.

[O'N66] B. O'Neill, "*Elementary Differential Geometry*", Academic Press, Orlando, Florida, 1966.

[OM95] A. Opalach and S. Maddock, "An Overview of Implicit Surfaces", *in "Introduction to Modelling and Animation Using Implicit Surfaces", Course Notes No. 3, Computer Graphics International*, Leeds, June 25-30, 1995, part 1, pp. 1-13, 1995.

[OW00] K. Otto and K. Wood, "*Product Design*", Prentice Hall, 2000.

[PC02] V. Pascucci and K. Cole-McLaughlin. "Efficient Computation of the Topology of Level Sets", In *Proceedings Visualization 2002*, 2002.

[Pope96] P. L. A. Popelier, "MORPHY, A Program for an Automatic Atoms in Molecules Analysis", *Computer Physics Communication*, Vol. 93, pp. 212-240, 1996.

[Pope98] P. L. A. Popelier, "MORPHY98 — A Program Written by P. L. A. Popelier with a Contribution from R. G. A. Bone", *UMIST*, Manchester, England, 1998.

[RHVKJ00] Z. Rusak, I. Horvath, J. S. M. Vergeest, G.Kuczogi, and J. Jansson, "First Result of the Development of Particle System Modelling for Shape Conceptualization", *International Design conference –Design 2000*, Dubrovnik, May 23-26, 2000.

[Sher98] A. Sherstyuk, “Convolution Surface in Computer Graphics”, *Ph.D. Dissertation, Monash University*, Australia, 1998.

[Sher99] A. Sherstyuk, “Shape Design Using Convolution Surfaces”, Proceedings of Shape Modeling International '99, March 1999.

[SKK91] Y. Shinagawa, T.L. Kunii, and Y.L.Kergosien, “Surface Coding Based on Morse Theory*”, IEEE Computer Graphics and Application* Vol. 11, No. 5, pp.66-78, 1991.

[Sier99] D. Siersma, “Voronoi Diagrams and Morse Theory of the Distance Function”, In *Geometry in Present Day Science*, O.E. Barndor-Nielsen and E.B.V. Jensen (eds.), World Scientific, pp.187-208, 1999.

[SH98] B. T. Stander and J. C. Hart, “Guaranteeing the Topology of an Implicit Surface Polygonization for Interactive Modeling”, *Proceeding of Implicit Surfaces 98, Eurographics/SIGGRAPH Workshop*, pp. 69-76, June 1998.

[SP95] R. Storn and K. Price, “Differential Evolution – a Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces”, *Technical Report TR-95-012*, ICSI, 1995.

[SP96] R. Storn and K. Price, "Minimizing the Real Functions of the ICEC'96 Contest by Differential Evolution", *IEEE Conference on Evolutionary Computation*, pp. 842 – 844, Nagoya, 1996.

[SV03] A. Szymczak, and J. Vanderhyde, “Extraction of Topologically Simple Isosurfaces from Volume Datasets”, *IEEE Visualization 2003*, pp. 67-74, 2003.

[TH03] R. Tam and W. Heidrich, “Shape Simplification Based on the Medial Axis Transform”, *IEEE conference on Visualization,* Seattle, Oct. 19-24, Washington, USA, 2003.

[Thom00] D. C. Thompson, "Feasibility of a Skeletal Modeler for Conceptual Mechanical Design", *Ph.D. dissertation, The University of Texas at Austin*, 2000.

[TO02] G. Turk and J. F. O'Brien, “Modelling with Implicit Surfaces that Interpolate”. *ACM Transaction on Graphics*, Vol. 21, No. 4, pp. 855-873, 2002.

[Van95] C. G. C. Van Dijk, “New Insights in Computer-Aided Conceptual Design*”, Journal of Design Studies*, Vol. 16, No. 1, pp. 62-80, January 1995.

[VGW94] A. Van Gelder and J. Wilhelms, “Topological Considerations in Isosurface Generation”, *ACM Transactions on Graphics*, Vol. 13, No. 4, pp. 337-375, October 1994.

[VVBS97] M. Van Kreveld, R. Van Oostrum, C. L. Bajaj, and D. L. Schikore, "Contour Trees and Small Seed Sets for Isosurface Traversal", In *Proceedings of the 13th Annual Symposium on Computational Geometry*, pp. 212-220, June 1997.

[VKSM04] G. Varadhan, S. Krishnan, T. Sriram, and D. Manocha, "Topology Preserving Surface Extraction using adaptive subdivision", *Technical report*, Department of Computer Science, University of North Carolina, 2004.

[Weis07] E. W. Weisstein, "Topology", *http://mathworld.wolfram.com/Topology.html*.

[WS01] G. Wesche and H. Seidel, "FreeDrawer: a Free-Form Sketching System on the Responsive Workbench", *Proceedings of the ACM symposium on Virtual reality software and technology*, pp. 167-174, Baniff, Alberta, Canada, November 15-17, 2001.

[WHDS02] Z. Wood, H. Hoppe, M. Desbrun, and P. Schröder, "Isosurface Topology Simplication", *Technical Report MSR-TR-2002-28,* Microsoft Research, 2002.

[WHDS04] Z. Wood, H. Hoppe, M. Desbrun, and P. Schröder, "Removing Excess Topology from Isosurfaces", *ACM Transactions on Graphics*, Vol. 23, No. 2, pp. 190-208, April 2004.

[WM99] S. T. Wu and M. G. Malheiros, "On Improving the Search for Critical Points of Implicit Functions"**,** *Proceedings of The Fourth International Workshop in Implicit Surface*. 1999.

[WMW86] G. Wyvill, G. C. McPheeters, and B. Wyvill, "Data Structure for Soft Objects", *The Visual Computer*, Vol. 2, No. 4, pp. 227-234, 1986.

[webCGAL07] Computational Geometry Algorithms Library, URL: *www.cgal.org*.

[webDE07] Differential Evolution (DE), URL: *www.icsi.berkeley.edu/~storn/code.html*.

[webHyper07] Hyperfun Project, URL: *www.hyperfun.org*.

Vita

Guohua Ma was born on September 30, 1970 in Shijiazhuang HeBei, China. She is the daughter of Daming Ma and Baorong Lü. She received the Bachelor of Science degree in Mechanical Engineering from XIDIAN University, China, in 1992, and her Master of Science in Mechanical Engineering from National University of Singapore in 1999.

Permanent address: 416 Qingyuan Street, #3-4-301
Shijiazhuang, Hebei, 050021
P. R. China
guo_hua_ma@hotmail.com

This dissertation was typed by the author.